(19) JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11) Publication number 2000222207 A

(43) Date of publication of application: 11.08.00

(51) Int. Cl G06F 9/38

(21) Application number 2000026002

(22) Date of filing: 03.02.00

(30) Priority: 03.02.99 US 99 243721

(71) Applicant: INTERNATL BUSINESS MACH CORP <IBM>

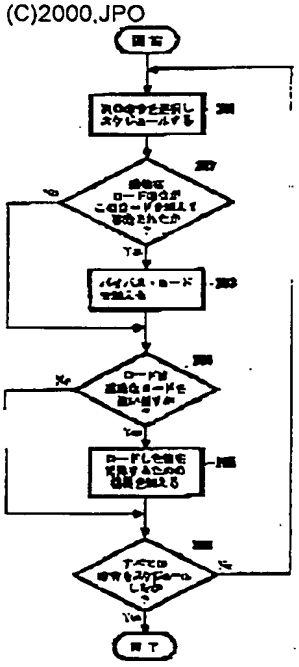
(72) Inventor: MICHAEL K GUSUWINDO

(54) METHOD AND DEVICE FOR CHANGING ORDER OF LOAD OPERATION OF COMPUTER PROCESSING SYSTEM

(57) Abstract:

PROBLEM TO BE SOLVED: To change the order of storing operation by passing data to which an unselected instruction gained read access before to a selected instruction when the address that the unselected instruction accessed is the same as the address that the selected instruction is to access.

SOLUTION: A next instruction is selected and scheduled for execution (S201). When the selected instruction is a load instruction, it is decided whether or not an unselected instruction which may ambiguously refer to the same storage location has been selected and moved beyond the load instruction (S202). When the selected instruction is the load instruction and it is decided that the unselected instruction which may ambiguously refer to the same storage location has been moved beyond the selected load instruction, operation for eliminating the ambiguity is performed (S203). These include a test of the selected load instruction for determining whether or not there is interference between load instructions out of order and preparations for a bypass sequence to be executed in case of the interference.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2000－222207
（P2000－222207A）
(43)公開日　平成12年８月11日(2000.8.11)

(51)Int.Cl.[7]	識別記号	ＦＩ	テーマコード（参考）
Ｇ０６Ｆ　9/38	３５０ ３１０	Ｇ０６Ｆ　9/38	３５０Ａ ３１０Ｆ

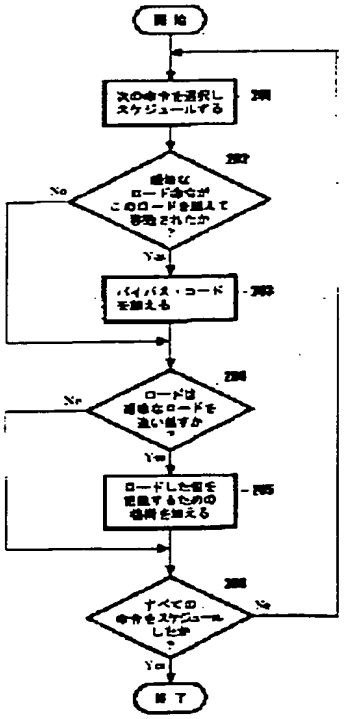
審査請求　有　　請求項の数37　ＯＬ　（全 28 頁）

(21)出願番号　特願2000－26002(P2000－26002)
(22)出願日　平成12年２月３日(2000.2.3)
(31)優先権主張番号　０９／２４３７２１
(32)優先日　平成11年２月３日(1999.2.3)
(33)優先権主張国　米国（ＵＳ）

(71)出願人　390009531
インターナショナル・ビジネス・マシーンズ・コーポレーション
ＩＮＴＥＲＮＡＴＩＯＮＡＬ　ＢＵＳＩＮＥＳＳ　ＭＡＳＣＨＩＮＥＳ　ＣＯＲＰＯＲＡＴＩＯＮ
アメリカ合衆国10504、ニューヨーク州　アーモンク　（番地なし）
(72)発明者　マイケル・ケー・グスウィンド
アメリカ合衆国06811　コネティカット州、ダンバリィ、　バダナラム・アヴェニュー　38　＃15エー
(74)代理人　100086243
弁理士　坂口　博　（外１名）

(54)【発明の名称】　コンピュータ処理システムにおけるロード動作を順序変更する方法および装置

(57)【要約】　（修正有）
【課題】コンピュータ処理システムにおけるロード動作を順序変更する方法および装置を提供する。
【解決手段】次の命令を選択し、命令シーケンス中のより早い位置に移動し、非選択命令が以前に選択された命令を越えて移動されており、非選択命令がデータを読み出すアドレスが、選択された命令がデータを読み出すアドレスと同じである場合、非選択命令によって以前に読み出されたデータを選択された命令にパスするバイパス・シーケンスを設け、非選択命令による将来の参照のために、選択された命令のレコード格納するための機構を追加する。



【特許請求の範囲】
【請求項１】アウトオブオーダ実行を実現するコンピュータ処理システムで命令を実行のためにスケジュールする方法であって、
次の命令を選択し、命令シーケンス中のその現在位置からより早い位置に移動するステップと、
選択命令が読み取りアクセスのために前記記憶場所を参照できるかどうかを決定するステップと、
選択命令が読み取りアクセスのために前記記憶場所を参照できる場合、読み取りアクセスのために前記記憶場所を曖昧に参照できる非選択命令が選択命令を越えてあらかじめ移動されたかどうかを決定するステップと、
非選択命令が選択命令を越えてあらかじめ移動されており、非選択命令がデータを読み取りアクセスした記憶場所のアドレスが、選択命令によってデータが読み取りアクセスされる記憶場所のアドレスと同じである場合、
選択命令の実行中に実行され、
選択命令に非選択命令によってあらかじめ読み取りアクセスされたデータをパスするバイパス・シーケンスを設けるステップと、
選択命令が読み取りアクセスのために記憶場所を参照できる場合、選択命令が以前に非選択命令を越えて移動されたかどうかを決定するステップと、
非選択命令による将来の参照のために、選択命令の記録を格納するための機構を追加するステップと、を含むことを特徴とする方法。
【請求項２】選択命令が非選択命令を越えてあらかじめ移動された場合、前記追加するステップを実行する請求項１記載の方法。
【請求項３】非選択命令が読み取りアクセスのための前記記憶場所を曖昧参照できる場合、前記追加するステップを実行する請求項１記載の方法。
【請求項４】選択すべき命令が残っているかどうかを決定するステップと、
選択するべき命令が残っている場合、前記選択するステップに戻るステップと、をさらに含む請求項１記載の方法。
【請求項５】異なる命令を越えてあらかじめ移動された第一の命令に、異なる命令を識別するロードタグ識別子を割り当てるステップをさらに含む請求項１記載の方法。
【請求項６】非選択命令が以前に選択命令を越えて移動されたかどうかを決定する前記ステップが、
ロードタグ識別子が非選択命令に割り当てられたことがあるかどうかを決定するステップと、
ロードタグ識別子が選択命令を識別するかどうかを決定するステップと、を含む請求項５記載の方法。
【請求項７】選択命令が以前に非選択命令を越えて移動されたかどうかを決定する前記ステップが、
ロードタグ識別子が選択命令に割り当てられたことがあるかどうかを決定するステップと、
ロードタグ識別子が非選択命令を識別するかどうかを決定するステップと、を含む請求項５記載の方法。
【請求項８】ロードタグ識別子をアウトオブオーダ・ロード命令語とともに格納するステップをさらに含む請求項５記載の方法。
【請求項９】アウトオブオーダ実行を実現するコンピュータ処理システムで命令を実行のためにスケジュールする方法であって、次の命令を選択し、命令シーケンス中のその現在位置からより早い位置に移動するステップと、
選択命令が読み取りアクセスのために記憶場所を参照するかもしれないかどうかを決定するステップと、
選択命令が読み取りアクセスのために記憶場所を参照するかもしれない場合、選択命令が、読み取りアクセスのために記憶場所を参照するかもしれない非選択命令に対してインオーダ命令であるかどうかを決定するステップと、
選択命令が、非選択命令に対してインオーダ命令であり、非選択命令がデータを読み取りアクセスしたところの記憶場所のアドレスが、選択命令がデータを読み取りアクセスするところの記憶場所のアドレスと同じである場合、前記選択命令の実行中に実行される、非選択命令によって以前に読み取りアクセスされたデータを選択命令にパスするバイパス・シーケンスを設けるステップと、
選択命令が読み取りアクセスのために記憶場所を参照するかもしれない場合、前記選択命令が非選択命令に対してアウトオブオーダ命令であるかどうかを決定するステップと、
前記選択命令が前記非選択命令に対してアウトオブオーダ命令である場合、前記非選択命令による将来の参照のために、選択命令のレコードを格納するための機構を追加するステップと、を含むことを特徴とする方法。
【請求項１０】選択命令が非選択命令に対してアウトオブオーダ命令である場合、前記追加ステップを実行する請求項９記載の方法。
【請求項１１】非選択命令が読み取りアクセスのために記憶場所を曖昧に参照するかもしれない場合、前記追加ステップを実行する請求項９記載の方法。
【請求項１２】レコードが、選択命令がデータを読み取りアクセスしたところの記憶アドレスと、データに対応する値とを含む請求項９記載の方法。
【請求項１３】各アウトオブオーダ命令に識別子を割り当てて、アウトオブオーダ命令およびアウトオブオーダ命令に対してインオーダである命令を識別するステップをさらに含む請求項９記載の方法。
【請求項１４】識別子をアウトオブオーダ命令語の中に格納する請求項１３記載の方法。
【請求項１５】識別子を、それに割り当てられたアウト

オブオーダ命令のレコードの中に記録する請求項１３記載の方法。

【請求項１６】アウトオブオーダ命令に割り当てられた識別子で識別されるインオーダ命令が実行されており、干渉しないものであると決定されている場合、アウトオブオーダ命令のレコードを破壊するための手段を追加するステップをさらに含む請求項１３記載の方法。

【請求項１７】アウトオブオーダ命令のレコードが後の割り当て要求によって自動的に取って代わられるように設ける請求項９記載の方法。

【請求項１８】レコード中に格納されたデータ単位が、不可分性を保証する可能な最大データ単位である請求項９記載の方法。

【請求項１９】多数のアウトオブオーダ命令がインオーダ命令と同じアドレスを参照する場合、最下位の論理プログラム・アドレスを有するデータがインオーダ・ロード命令にパスされることを保証するために優先順位符号化を実現するステップをさらに含む請求項９記載の方法。

【請求項２０】命令シーケンスが少なくとも一つの処理装置によって実行され、多数の読み取りアクセスがアウトオブオーダ・ロード命令および少なくとも一つの他のロード命令によって起動されるコンピュータ処理システム中の記憶場所からの多数の読み取りアクセスの間に得られるデータ値の間の整合性を執行する方法であって、

アウトオブオーダ・ロード命令を実行して少なくとも一つの処理装置を制御して、アウトオブオーダ・ロード命令によって識別される記憶場所から第一のデータを少なくとも読み取りアクセスさせるステップと、

少なくとも一つの他のロード命令が使用するための、第一のデータがロードされたところの記憶場所のアドレスと、第一のデータに対応する値とを含む、アウトオブオーダ・ロード命令のレコードを生成するステップと、

少なくとも一つの他のロード命令を実行し、少なくとも一つの他のロード命令の実行中に少なくとも一つの処理装置を制御して、

アウトオブオーダ命令が第一のデータをロードしたところの記憶場所のアドレスが、少なくとも一つの他のロード命令がデータをロードするところのアドレスと重複する、または同じであるかどうかを決定するステップと、

アドレスが同じである、または重複する場合、それぞれ第一のデータまたはその一部をレコードから少なくとも一つの他のロード命令にパスするステップと、を実行させるステップと、を含むことを特徴とする方法。

【請求項２１】処理装置が命令シーケンス中のアウトオブオーダ・ロード命令の元の位置に達した場合、レコードを破壊するステップをさらに含む請求項２０記載の方法。

【請求項２２】命令シーケンス中のアウトオブオーダ・ロード命令の元の位置でレコードを削除するための手段を追加するステップをさらに含む請求項２０記載の方法。

【請求項２３】各アウトオブオーダ命令に識別子を割り当てて、アウトオブオーダ命令およびアウトオブオーダ命令に対してインオーダである命令を識別するステップをさらに含む請求項２０記載の方法。

【請求項２４】アウトオブオーダ命令を実行する前記ステップが、少なくとも一つの処理装置を制御して、第一のデータをアウトオブオーダ命令によって識別される第一の標的レジスタ中に配置させるステップをさらに含む請求項２０記載の方法。

【請求項２５】前記決定ステップおよびパスするステップが、少なくとも一つの処理装置によって実行される一つの命令を含む請求項２０記載の方法。

【請求項２６】命令シーケンスが少なくとも一つの処理装置によって実行され、多数の読み取りアクセスが少なくとも一つのアウトオブオーダ・ロード命令および一つのインオーダ・ロード命令によって起動されるコンピュータ処理システムの記憶サブシステム中の場所からの多数の読み取りアクセスの間に得られるデータ値の間の整合性を執行するための装置であって、

データが読み取りアクセスされたところの記憶サブシステム中の場所のアドレスと、データに対応する値とを含む、少なくとも一つのアウトオブオーダ・ロード命令のレコードを格納するための格納手段と、

インオーダ命令を処理するとき、少なくとも一つのアウトオブオーダ・ロード命令がデータを読み取りアクセスしたところの記憶サブシステム中の場所のアドレスと、インオーダ・ロード命令がデータを読み取りアクセスするところの記憶サブシステム中の場所のアドレスとを比較するための比較論理と、

前記格納手段に指図してレコードを格納させ、少なくとも一つのアウトオブオーダ命令のレコードにアクセスしてアドレスを前記比較論理に供給するための制御論理と、

前記比較論理によって比較されたアドレスがそれぞれ等しい、または重複する場合、前記制御論理の制御の下で、前記格納手段に格納された値またはその一部をインオーダ命令に供給するためのデータ選択機構と、を含むことを特徴とする装置。

【請求項２７】前記格納手段が、ＣＡＭ構造またはレコードを格納するために予め指定された前記記憶サブシステム中の区域である請求項２６記載の装置。

【請求項２８】前記レコードが、前記アウトオブオーダ・ロード命令に対応するインオーダ・ロード命令を識別するための識別子をさらに含む請求項２６記載の方法。

【請求項２９】前記識別子で識別されるように前記アウトオブオーダ命令に対応する前記インオーダ命令が実行されており、干渉しないものであると決定されている場合、記格納手段に格納されたレコードを削除するための

手段をさらに含む請求項２８記載の装置。
【請求項３０】少なくとも一つのアウトオブオーダ・ロード命令およびインオーダ・ロード命令を処理するとき、データを読み取りアクセスするところの記憶サブシステム中の場所のアドレスを生成するためのアドレス生成論理をさらに含む請求項２６記載の装置。
【請求項３１】インオーダ・ロード命令がデータを読み取りアクセスするところの記憶中の場所のアドレスからデータを読み取りアクセスしたことがある多数のアウトオブオーダ・ロード命令が存在する場合、命令シーケンス中で最下位の元の位置を有するデータを検出するための優先順位符号化手段をさらに含む請求項２６記載の装置。
【請求項３２】命令シーケンスが少なくとも一つの処理装置によって実行され、多数の読み取りアクセスが少なくとも一つの先に実行されたロード命令および現在実行中のロード命令によって起動されるコンピュータ処理システムの記憶サブシステム中の場所からの多数の読み取りアクセスの間に得られるデータ値の間の整合性を執行するための装置であって、
データが読み取りアクセスされたところの記憶サブシステム中の場所のアドレスと、データとを含む、少なくとも一つの先に実行された命令のレコードを格納するための格納手段と、
現在実行中の命令を処理するとき、少なくとも一つの先に実行された命令がデータを読み取りアクセスしたところの場所のアドレスと、現在実行中の命令がデータを読み取りアクセスするところの場所のアドレスとを比較し、少なくとも一つの先に実行された命令が、現在実行中の命令よりも後の命令番号をそれに対応して有するかどうかを決定するための干渉試験および優先順位エンコーダと、
前記格納手段に指図してレコードを格納させ、少なくとも一つの先に実行された命令のレコードにアクセスしてアドレスを前記干渉試験および優先順位エンコーダに供給するための制御論理と、
前記前記干渉試験および優先順位エンコーダによって比較されたアドレスが等しい、または重複し、少なくとも一つの先に実行された命令が、より遅い命令番号をそれに対応して有する場合、前記干渉試験および優先順位符号化論理の制御の下で、前記格納手段に格納されたデータまたはその一部を現在実行中の命令に供給するためのデータ選択機構と、を含むことを特徴とする装置。
【請求項３３】前記干渉試験および優先順位エンコーダによって比較されるアドレスが等しくなく、重複もしない場合、前記データ選択機構が、記憶サブシステム中に格納されたデータを現在実行中の命令に供給する請求項３２記載の装置。
【請求項３４】少なくとも一つの先に実行された命令が二つ以上存在する場合、前記データ選択機構が、論理的にもっとも早い干渉する先に実行された命令に対応するデータを格納手段から現在実行中の命令に供給する請求項３２記載の装置。
【請求項３５】現在実行中の命令がデータを読み取りアクセスするところの記憶中の場所のアドレスからデータを読み取りアクセスしたことがある少なくとも一つの先に実行された命令が二つ以上存在する場合、前記干渉試験および優先順位エンコーダが、命令シーケンス中で最下位の元の位置を有するデータを検出する請求項３２記載の装置。
【請求項３６】命令番号が、処理装置によって所与の期間に処理することができる数の少なくとも２倍の数ｎの命令を符号化するのに十分に広い請求項３２記載の装置。
【請求項３７】少なくとも一つの先に実行された命令の命令番号が、現在実行中の命令の命令番号に続くｎ－１の範囲にある場合、少なくとも一つの先に実行された命令が現在実行中の命令よりも後の命令番号を有する請求項３６記載の装置。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、一般にコンピュータ処理システムに関し、特に、コンピュータ・プログラムにおけるロード動作を順序変更する方法および装置に関する。本発明は、プログラムが生成されるときに順序変更される動作（静的順序変更）および実行時に順序変更される動作（動的順序変更）に適用可能である。
【０００２】
【従来の技術】現在の高性能プロセッサは、プログラムにおける命令レベルの並列性を利用するため（すなわち、一度に二つ以上の命令を実行するため）、スーパスケーラ技術、スーパパイプライニング技術および／またはＶＬＩＷ（長大命令語）技術に依存する。一般に、これらのプロセッサは、多数の機能装置を含み、命令シーケンスを実行し、１サイクルあたり二つ以上の命令を記憶から取り出すことができ、従属性および資源可使性に応じて１サイクルあたり二つ以上の命令を実行にディスパッチすることができる。
【０００３】所与の時点でディスパッチされる命令をプロセッサが選択するところの命令のプールは、アウトオブオーダ実行の使用によって拡大される。アウトオブオーダ実行とは、後から現れる動作が、その動作によって要求される資源が使用中でないならば先に実行されるよう、命令シーケンス中の動作を順序変更する技術である。したがって、アウトオブオーダ実行は、多数の機能装置の可使性を利し、さもなくばアイドル状態になるであろう資源を活用することにより、プログラム全体の実行時間を短縮する。動作の実行の順序変更は、プログラムの機能的挙動が、命令を元の順序で実行したならば得られるであろうものと同じになるよう、それらの動作に

よって生じる結果をも順序変更することを要する。

【０００４】記憶関連の動作の場合、記憶ロード動作が記憶からデータを読み取り、それをプロセッサのレジスタにロードし、ロードしたデータに従属する動作のシーケンスを頻繁に実行する。したがって、アイドル状態の資源を使用することに加えて、記憶ロード動作の早めの（アウトオブオーダ）開始が、潜在的なキャッシュ・ミスを含む、記憶にアクセスする際の遅延を隠すことができる。

【０００５】一般に、アウトオブオーダ実行および結果の順序変更を実現するのには二つの基本的な手法がある。すなわち、動的順序変更および静的順序変更である。動的順序変更では、命令は実行時に解析され、命令および結果はハードウェア中で順序変更される。静的順序変更では、プログラムが生成されるとき、コンパイラ／プログラマが命令およびそれらの命令によって生じる結果を解析し、順序変更する。したがって、順序変更のタスクはソフトウェアによって達成される。これら二つの手法は、組み合わせて実現することもできる。

【０００６】一般にはアウトオブオーダ実行、特には記憶間動作をサポートするための有意な研究が実施されてきたが、そのような研究は主としてユニプロセッサ実行に集中してきた。これは逆に、１個のプロセッサ上で実行するように設計された一つの命令ストリームにおけるロードおよび（同期）ストア動作の間の順序付けに焦点を置いていた。本発明は、多重処理環境で典型的に見られる非同期記憶参照および１個の記憶セルに対する多数の読み取り動作間の順序変更に対するそれらの影響の問題を扱う。このような変換（すなわち順序変更）は、厳密なユニプロセッサ環境では安全であるが、マルチプロセッサ環境は、さらなる問題点、たとえば、別個の未知の命令ストリームを有する別のプロセッサによって書き込みが実施される危険性を提起する。

【０００７】予測可能で反復可能なプログラムの計算を達成するため「シーケンス整合性」の要件が、L. Lamportによる文献「How to Make a Multiprocessor thatCorrectly Executes Multiprocess Programs」IEEE Transactions on Computers, C-28(9), pp. 690-91（１９７９年９月）に記載されている。Lamportによるこの文献は、マルチプロセッサ・システムを「実行の結果が、すべのプロセッサの動作を順序どおりに実行したときと同じであり、個々のプロセッサの動作が、そのプログラムによって指定された順序でこのシーケンス中に現れる」ならば、シーケンス整合したものと定義している。静的な投機実行の場合、順序変更されたプログラム・テキストではなく、元の論理的プログラム・テキストの順序が権威を有し、コンパイラおよびハードウェア実現形態が共同してその元の順序に等しい実行を生成しなければならない。

【０００８】システム中の多数のプロセッサの間のコヒーレンス・プロトコルを簡素化しながら適切な性能を達成するためには、上述した厳密にシーケンス整合した順序のいくらかの緩和が可能である。許容しうるタイプの順序変更は、特定の実現形態によって保証される記憶整合性モデルに依存する。現在使用され、提唱されている整合性モデルおよびそれらの特性の概要が、S. AdveおよびK. Gharachorlooによる文献「Shared Memory Consistency Models: A Tutorial」TechnicalReport 9512, Dept. of Electrical and Computer Engineering, Rice University, Houston, Tx（１９９５年９月）に記載されている。

【０００９】これらすべての緩和モデルにおける基本的要件は、記憶コヒーレンスを達成するための書き込み直列化、すなわち、同じ場所に対するすべての書き込みが何らかの順序で直列化され、いかなるプロセッサに関してもその順序で実行されるということである。これは、Lamportによって記載された、各記憶セルを記憶モジュールとみなすシーケンス整合性に等しい。われわれは、１個の記憶セルに関するシーケンス整合性を書き込み直列とみなして、より大きな記憶モジュールの場合のシーケンス整合性とは区別する。記憶コヒーレンスは、同じ記憶場所に対する連続的なロード動作がその記憶場所で提示されるデータ項目の弱昇順を保存することを保証することによって達成することができる。したがって、ロード動作のシーケンスでは、いかなるロードも、その前のロードと同じ、またはそれよりも後のデータ項目しか提示することができない。

【００１０】たとえば、第二のプロセッサによって所与の記憶場所に書き込まれるデータ項目ｄ１、ｄ２、ｄ３、ｄ４、ｄ５、ｄ６などのシーケンスを考えてみよう。第一のプロセッサによるその記憶場所からの連続的なロード動作は、第一のロード動作によって戻されたものと同じデータ項目またはその記憶セル中に存在する、より後の項目を提示することができる。したがって、第一のロード動作がデータ項目ｄ２を戻したならば、第二のロード動作は、データ項目ｄ２、ｄ３、ｄ４などを戻すことはできるが、データ項目ｄ１を戻すことはできない。あるいはまた、第一のロード動作がデータ項目ｄ４を戻したならば、第二のロード動作は、データ項目ｄ４、ｄ５、ｄ６などを戻すことはできるが、データ項目ｄ１、ｄ２またはｄ３を戻すことはできない。

【００１１】直列実行では、この問題が、順方向に進む時間の性質によって自動的に解決されることは明らかである。しかし、アウトオブオーダ・プロセッサに関しては、同じ記憶場所にアクセスするロード動作がアウトオブオーダになって、静的に後のロード命令が、より後の時点で実行されるデータ項目を、データ項目のシーケンスの中でその静的に先行するロード命令よりも先に読み出すことになる。

【００１２】動作を順序変更する能力を制限する一つの

要因は、曖昧な記憶参照である。これは、記憶ロード動作が命令シーケンス中の別の記憶ロード動作よりも後で現れ、アウトオブオーダ動作によってアクセスされる記憶場所とインオーダ・ロード動作によってアクセスされる記憶場所とが異なるかどうかを前もって決定することができない場合に当てはまる。たとえば、以下のコード断片を考えてみよう。

【００１３】s=*(X+a*4+5)
u=s+4
t=*Y
v=t+8

*は、指定されたアドレスに対する記憶アクセスを示し、よって、*Yは、そのアドレスがYに含まれる記憶場所を示し、*（Ｘ＋ａ*４＋５）は、そのアドレスが式Ｘ＋ａ*４＋５によって指定される記憶場所を示す。

【００１４】ａがプロセッサのレジスタｒ１に格納された値であり、ＸおよびＹがレジスタｒ２およびｒ９の中にあり、ｓ、ｔ、ｕおよびｖがそれぞれレジスタｒ４、ｒ５、ｒ６およびｒ７に割り当てられていると仮定すると、上記コード断片を以下の命令シーケンスによって表現することができる（命令の名前の後の最初のレジスタが標的レジスタであり、残りのレジスタはオペランドである）。

【００１５】

```
mul     r10,r1,4      ;r10=a*4
add     r11,r10,5     ;r11=a*4+5
add     r12,r11,r2    ;r12=X+a*4+5
load    r4,(r12)      ;s=*(X+a*4+5)
add     r6,r4,4       ;u=s+4
load    r5,(r9)       ;t=*Y
add     r7,r5,8       ;v=t+8
```

【００１６】Ｘ＋ａ*４＋５およびＹが異なるアドレスを参照していると判断することができるならば、並列実行のために四つの式をスケジュールして、たとえば以下のシーケンスを出すことができる（記号||は並列実行を表す）。

【００１７】

```
mul     r10,r1,4      ||  load  r5,(r9)
add     r11,r10,5     ||  ...
add     r12,r11,2     ||  add       r7,r5,8
load    r4,(r12)      ||  ...
...                   ||  ...
add     r6,r4,4       ||  ...
```

【００１８】２個の実行装置を有する機械では、上記シーケンスは、完了するのに６サイクルを要する（ロードが２サイクルを要し、他の各動作が１サイクルずつ要すると仮定）。

【００１９】他方、Ｘ＋ａ*４＋５およびＹが常に異なると判断することができない（すなわち、アドレスが曖昧である）ならば、二つの式を元の順序でスケジュールしなければならず、完了するのに９サイクルを要する（同じく、ロードが２サイクルを要し、他の各動作が１サイクルずつ要すると仮定）。

【００２０】両方のロード・アドレスが同じ記憶場所を参照し、その記憶場所が、ｄ１の後にｄ２が続くデータ項目シーケンスを受けるならば、変数ｓおよびｔに関して全部で四つの読み取りアクセスの組み合わせが可能である。これらのうち、以下に示す最初三つの組み合わせは書き込み直列化されているが、四番目の組み合わせは書き込み直列化の要件を満たしていない。

【００２１】

	書き込み直列	書き込み直列	書き込み直列	非書き込み直列
第一のロード動作（ｓ）	ｄ１	ｄ１	ｄ２	ｄ２
第二のロード動作（ｔ）	ｄ１	ｄ２	ｄ２	ｄ１

【００２２】以下の例で、両方のロード動作が同じ記憶場所を参照するならば、順序変更された命令のスキームが、変数ｓの後に変数ｔを読み出す実際のユーザ・プログラムをして、後のデータ項目ｄ２が先のデータ項目に先行するシーケンスを見させることがわかる。これは、マルチプロセッサの同期化、すなわちＤＭＡデバイスとの通信に関して有意な問題である。

【００２３】第二のプロセッサが、ｓおよびｔへのロードによって参照される記憶場所にデータｄ１を書き込む。

```
mul     r10,r1,4      ||      load  r5,(r9)
add     r11,r10,5     ||      ...
```

第二のプロセッサがデータをｄ２に変更する。

```
add     r12,r11,r2    ||  add       r7,r5,8
load    r4,(r12)      ||  ...
...                   ||      ...
add     r6,r4,4       ||  ...
```

【００２４】これは、正味には、ｄ２をｓにロードし、ｄ１をｔにロードする効果を有し、それは、第二のプロ

セッサによって両方のロード動作によってアクセスされる記憶場所に実際に格納された、書き込み直列化された数値のシーケンスと整合しない。

【００２５】上記の例は不定形的である。記憶参照における曖昧さが、そうでなければ並列に実行することができるであろう動作の逐次実行を強要することにより、システム性能をひどく低下させる。しかし、ユーザ・プログラムによって認知されるロード結果値のシーケンスが書き込み直列化される限り、そのような命令の直列化を避けることができる（すなわち、論理的に後続するロード動作を論理的に先行するロード動作よりも先に実行することができる）。したがって、ロード動作のデータ・シーケンスがそれらの元のプログラム順序で記憶中の対応するデータ・シーケンスと整合している限り、インオーダ・ロード動作よりも先に実行されるアウトオブオーダ動作は有効である（すなわち、後続する各ロードが、論理的に先行するすべての読み取り動作と同じ値またはそれらに対して時系列的に後で生じる値を返す）。そのうえ、これらの値が整合しているならば、アウトオブオーダでロードされたデータに従属する動作もまた、アウトオブオーダで実行することができる。他方、これらの値が整合していないならば、アウトオブオーダでロードされたデータおよびそれから誘導される結果は無効であり、インオーダ点でロード動作および対応する従属動作を再び実行しなければならない。

【００２６】プロセッサによる曖昧な参照を伴う記憶動作を順序変更する問題の解消に向けて種々の試みがなされてきた。これらのスキームの大部分は、命令が静的に（すなわち、プログラムが生成されるとき）順序変更されると仮定している。これらのスキームはすべて、アドレス比較またはロード結果値比較のいずれかによる干渉の検出に依存する。干渉が検出されるならば、インオーダ・ロード動作によりも前に実行されるようにスケジュールされたアウトオブオーダ・ロード動作（および、場合によっては、すでに実行されたロードに従属する動作）を、その元のインオーダ点で再実行しなければならない。すなわち、これらの機構が、すべての干渉するロード動作をインオーダで再実行することにより、書き込み直列化を執行する。干渉検出および再実行は、余計な命令（ソフトウェアベースのスキーム）またはソフトウェア支援を要することもある専用ハードウェア資源（ハードウェアベースのスキーム）のいずれかによって実行される。

【００２７】既存の機構は、アドレスが重複するときの正しさを保証するため、先に実行されたロード命令（すなわち、アウトオブオーダ・ロード命令）が別のロード命令（すなわち、インオーダ・ロード命令）に干渉したことを認識し、先に実行されたアウトオブオーダ・ロード命令およびすでに実行されたロード命令（すなわち、アウトオブオーダ・ロード命令）に従属する命令を再実行する。

【００２８】たとえば、先に記したコード断片は、以下のように変形することができる。

【００２９】

```
t=*Y
v=t+8
s=*(X+a*4+5)

u=s+4
if(Y=(X+a*4+5))  /*アドレスを比較*/
        t=*Y
        v=t+8
endif
```

【００３０】静的順序変更の場合、コンパイラ・プログラマによって生成される命令の順序は、提案される種々のスキームの間で異なる。普通、別のロード命令を越えて移動されたロード命令は、ロード動作を実行し、他のロード命令によって使用されるアドレスを監視し始める新たな命令（または命令シーケンス）によって取って代わられる。移動されたロード命令が元々位置していた場所を示し、干渉するストア動作の監視の範囲の終わりを決定するためには、別の命令（またはアウトオブオーダ・ロード命令中の命令フィールド）が使用される。

【００３１】動的順序変更の場合、種々のロード命令は、プログラム順序で、すなわち、第一のロード命令の後に第二のロード命令が続く順序でプロセッサに提示される。プロセッサが命令を順序変更し、静的順序変更の場合と同様、プロセッサは、第一のロード命令が、第二の、すでに実行されたアウトオブオーダ・ロード動作によって読み出された記憶場所にロードするかどうかを検出することができなければならない。たとえば、プロセッサは、そのロード命令をアウトオブオーダ動作として印し、アウトオブオーダ動作と他のロード動作との干渉を検出するための機構を準備し、干渉が検出されたときプロセッサの状態を回復し、アウトオブオーダ・ロード命令およびそのアウトオブオーダ・ロード命令に従属する他の命令を再実行しなければならない。

【００３２】記憶ロード動作を順序変更するときマルチプロセッサ環境で非同期記憶動作を扱う関連技術の概要を以下に記載する。

【００３３】干渉の検出および先に実行されたアウトオブオーダ・ロード命令の再発行に基づくアウトオブオーダ・ロード動作のサポート機構が、１９９７年３月３１日に出願された「Support for Out-Of-Order Execution of Loads and Stores in a Processor」と題する、本明細書の譲受人に譲渡される米国特許出願第０８／８２９，６６９号に開示されている。この機構は、各アウトオブオーダ・ロード動作のアドレスを、そのアウトオブオーダ・ロードの元のプログラム位置に達するまで、待ち行列（「ロードヒットロード待ち行

列」）に入れておく。他の（インオーダ）ロード動作が、ロードヒットロード待ち行列中の項目に対してそれらのアドレスを検証し、干渉が検出されるならば、干渉するアウトオブオーダ・ロード動作（およびすべての従属動作）がプロセッサによって再発行される。

【００３４】代替の検出機構が、ロード・データ検証に基づくロード／ストア干渉検出に関連して、１９９８年５月２６日に発行された「Method and Apparatus for Reordering Memory Operations in a Processor」と題する、本明細書の譲受人に譲渡された米国特許第５，７５８，０５１号に記載されている。この手法では、アウトオブオーダ・ロード動作によってアクセスされるデータ項目をインオーダで読み取り、そのインオーダ・ロード動作の結果をアウトオブオーダ結果と比較する。二つの値が同一であるならば、検出可能な干渉は起こっておらず、プログラムは実行を続ける。しかし、二つの値が同一ではないならば、インオーダ・ロード動作によって返された値を使用して、すべての従属命令を再実行する。ロード検証に基づく干渉検出の場合、干渉を検出することができなければ、干渉は存在しないと推定する。この手法は、干渉を監視するために必要なハードウェアの量および再実行の回数を減らすが、アウトオブオーダで移動されるロード動作ごとに第二のインオーダ・ロードを実行するためのさらなる帯域幅を要する。

【００３５】１９９７年４月２９日に発行された「Method and Apparatus for Reordering Memory Operations in a Superscalar or Very Long InstructionWord Processor」と題する、本明細書の譲受人に譲渡された米国特許第５，６２５，８３５号は、記憶動作の順序変更を記憶動作の投機実行と組み合わせている。記憶動作の順序変更は、

・命令レベルの並列性を利用するための、コンパイラによるコードの静的順序変更、

・記憶参照における競合を検出し、アウトオブオーダでロードされるデータを操作するための、特別なハードウェア・サポート、および

・アウトオブオーダでロードされたデータに対して演算を加え、競合の検出から回復するためのコンパイラ生成コードに依存する。

【００３６】特別なハードウェア・サポートは、アウトオブオーダで実行されるロード動作の結果の宛先となりうるレジスタごとのアドレス・レジスタと、そのような各アドレス・レジスタと対応する比較器とからなる。データをアウトオブオーダでロードし、そのようなデータおよびそれから導出される他の値をプログラム中のインオーダ点で「コミット」するために「特別な命令」を使用する。各アウトオブオーダ・ロードは、対応するアドレス・レジスタの中に、ロードしたデータの記憶アドレスおよびサイズを記録する。各ストア動作が、すべてのアドレス・レジスタの内容に対する（アドレス、サイズ）組の比較を起動する。マルチプロセッサ環境では、記憶明瞭化の役割を担う処理装置が、他のプロセッサによって発行されるすべての非同期ストア要求を受け、そのようなストア要求に干渉するすべてのアウトオブオーダ・ロード動作を無効にする。たとえば、対応するレジスタ中の（アドレス、サイズ）組が別のレジスタ中の別の（アドレス、サイズ）組にマッチするならば、その対応するアドレス・レジスタを無効と印す。インオーダ点で特別なコミット命令を実行して、対応するアドレス・レジスタが有効であるかどうかを検査し、有効であるならば、アウトオブオーダでロードされたデータと記憶中のデータとはコヒーレントである。他方、アドレス・レジスタが無効であるならば、アウトオブオーダでロードされたデータと記憶内容とはコヒーレントではない。したがって、ロード動作およびそれに従属する他の動作を再実行しなければならない。この時点でトラップが呼び出されて、実行制御を、コンパイラによって生成される回復コードに移し、それがロード動作および従属動作を再実行する。

【００３７】

【発明が解決しようとする課題】現在の方略（静的および動的なアウトオブオーダ実行のためのもの）は、動作のアウトオブオーダ実行と干渉を検出するためのスキーム（ソフトウェアまたはハードウェアで実現される）とを組み合わせることにより、書き込み直列化を扱う。ひとたび干渉が検出されたならば、これらのスキームは、プロセッサをして、干渉するアウトオブオーダ・ロード動作およびすべての従属動作を再実行させて、干渉する命令をインオーダで実行することによって書き込み直列な挙動を執行する。

【００３８】これが貴重な発行帯域幅を消費し、全体でより少ない数の実行しか完了させることができなくなる。加えて、コードを再実行するための機構を設けなければならない。静的スケジューリングの場合、検査コード、アウトオブオーダ・ロード命令およびその従属コードを含むコード断片が再びプログラム・テキストに配置され、そのような配置が暗示するすべてのコード・サイズおよび命令キャッシュ性能のペナルティを伴う。あるいはまた、動的実行の場合には、命令は、再発行バッファ中に維持され、再びディスパッチされる。

【００３９】以下のコード断片は、典型的なコンパイラによって生成されるようなYに対するアウトオブオーダ・アクセスで、静的に生成されるスケジュールにおいて同じコードを二度、プログラム・コード中に配置する方法を示す（複製コード部を「*」で示し、検査コードを「÷」で示す）。

【００４０】

```
v=t+8
s=*(X+a*4+5)
u=s+4
if(Y=(X+a*4+5))  /*アドレスを比較*/
        t=(*Y)
*
        v=t+8
*
endif
+
```

【００４１】この例からわかるように、静的コード生成の場合、コード・サイズのペナルティは非常に厳しい。

【００４２】動的に生成される命令スケジュールを使用するプロセッサ（たとえばスーパスケーラ・プロセッサ）に関しては、これらのプロセッサは、命令を再実行する決定を発行バッファに送り、リネームされたレジスタ・テーブル中の以前の項目を無効にするための論理を要する。

【００４３】干渉が検出されると、両機構が先に実行された命令を再実行しなければならないため、達成可能な性能に対するさらなるペナルティが生じる。したがって、いずれの手法も、他の命令の実行に使用することができたであろう発行スロットを使用することによって命令ストリームを再実行しなければならないことから、性能の低下を被る。ここまで論じたことから、順序変更された記憶読み取り参照の場合に干渉を修正するのに、再実行（すなわち、ひとたび違反状況が検出されるなら、ロード動作およびすべての従属動作を再実行すること）が効率的な方法ではないということになる。

【００４４】

【課題を解決するための手段】上述した課題ならびに関連する他の従来技術の課題は、本発明、すなわち、プロセッサにおける記憶動作を順序変更する方法および装置によって解決される。

【００４５】本発明の一つの態様で、アウトオブオーダ実行を実現するコンピュータ処理システムで命令を実行のためにスケジュールする方法は、次の命令を選択し、命令シーケンス中のその現在位置からより早い位置に移動するステップと、選択命令が読み取りアクセスのために記憶場所を参照するかもしれないかどうかを決定するステップと、選択命令が読み取りアクセスのために記憶場所を参照するかもしれない場合、読み取りアクセスのために記憶場所を曖昧に参照するかもしれない非選択命令が以前に選択命令を越えて移動されかどうかを決定するステップと、非選択命令が以前に選択命令を越えて移動されており、非選択命令がデータを読み取りアクセスしたところの記憶場所のアドレスが、選択命令がデータを読み取りアクセスするところの記憶場所のアドレスと同じである場合、選択命令の実行中に実行される、非選択命令によって以前に読み取りアクセスされたデータを選択命令にパスするバイパス・シーケンスを設けるステップと、選択命令が読み取りアクセスのために記憶場所を参照するかもしれない場合、選択命令が以前に非選択命令を越えて移動されたかどうかを決定するステップと、非選択命令による将来の参照のために、選択命令のレコードを格納するための機構を追加するステップと、を含む。

【００４６】本発明のもう一つの態様では、命令シーケンスが少なくとも一つの処理装置によって実行され、多数の読み取りアクセスがアウトオブオーダ・ロード命令および少なくとも一つの他のロード命令によって起動されるコンピュータ処理システム中の記憶場所からの多数の読み取りアクセスの間に得られるデータ値の間の整合性を執行する方法は、アウトオブオーダ・ロード命令を実行して少なくとも一つの処理装置を制御して、アウトオブオーダ・ロード命令によって識別される記憶場所から第一のデータを少なくとも読み取りアクセスさせるステップと、少なくとも一つの他のロード命令が使用するための、第一のデータがロードされたところの記憶場所のアドレスと、第一のデータに対応する値とを含む、アウトオブオーダ・ロード命令のレコードを生成するステップと、少なくとも一つの他のロード命令を実行し、少なくとも一つの他のロード命令の実行中に少なくとも一つの処理装置を制御して、アウトオブオーダ命令が第一のデータをロードしたところの記憶場所のアドレスが、少なくとも一つの他のロード命令がデータをロードするところのアドレスと重複する、または同じであるかどうかを決定するステップと、アドレスが同じである、または重複する場合、それぞれ第一のデータまたはその一部をレコードから少なくとも一つの他のロード命令にパスするステップと、を実行させるステップと、を含む。

【００４７】本発明のこれらの態様ならびに他の態様、特徴および利点は、添付図面と関連させて読まれる好ましい実施態様の以下の詳細な説明から理解されるであろう。

【００４８】

【発明の実施の形態】本発明は、コンピュータ処理システムにおけるロード動作を順序変更する方法および装置に関する。このようなコンピュータ処理システムでは、命令シーケンスは、少なくとも一つの処理装置によって

実行されるため、記憶に格納される。本発明は、プログラムが生成されるときに順序変更される動作（静的順序変更）および実行時に順序変更される動作（動的順序変更）に適用可能である。さらに、本発明は、ソフトウェアおよび　またはハードウェアベースの実現形態に適し、ユニプロセッサ・システム（たとえば、周辺装置が主記憶に直接アクセスするシステム）およびマルチプロセッサ・システムで適用可能である。

【００４９】本発明の明確な理解を容易にするため、まず、本明細書で使用する術語の定義を挙げる。「ロード命令」とは、記憶読み取りアクセスおよび場合によってはロードされた値に基づく計算を実行する命令をいう。たとえば、ロード命令は、たとえば、記憶場所からのデータをオペランドとして用いる論理、算術および他の命令を含むことができる。「アウトオブオーダ実行」とは、命令シーケンス中の動作を、後で現れる動作によって求められる資源が使用されていないならば、後で現れるその動作が先に実行されるよう、順序変更する技術である。たとえば、アウトオブオーダ・ロード命令は、ロード命令を命令シーケンス中のその元の位置から命令シーケンス中のより早い位置に移動させることにより、静的または動的に生成することができる。このようなアウトオブオーダ・ロード命令は、データを読み出すところの記憶中の場所と、そのデータを配置するところの第一の宛先レジスタとを識別する。「曖昧な記憶参照」とは、命令ストリーム中である記憶ロード動作が別の記憶ロード動作の後で現れ、それら二つの記憶ロード動作によってアクセスされる記憶場所が異なるかどうかを事前に決定することができない場合をいう。「投機動作」とは、たとえば、それが位置する実行経路がたどられないかもしれず、代わりに別の実行経路がたどられるかもしれないため、必ずしも実行されないかもしれないアウトオブオーダ動作をいう。

【００５０】以下の記載および対応する実施例は、二つの命令（別段記さない限り）、すなわち、第二の、論理的に先行する命令よりも先にアウトオブオーダで実行される第一の命令と、インオーダ命令と呼ばれるその第二の命令とに基づいて記載する。したがって、別段記さない限り、「インオーダ」という指定は、論理的に先行するインオーダ命令と、第一の「アウトオブオーダ」命令との順序的関係のみをいう。しかし、本発明は、多数のアウトオブオーダおよびインオーダのロード動作を包含するように拡張することが容易であることが当業者によって理解されよう。上記二つの命令（すなわち、インオーダおよびアウトオブオーダのロード命令）が第三の命令などに対してインオーダまたはアウトオブオーダであってもよいことを理解されたい。

【００５１】まず、本発明の概念を読者に紹介するため、本発明の概要を提供する。続いて、本発明の種々の態様のより詳細な説明を提供する。

【００５２】したがって、一般的に述べるならば、本発明は、データ・バイパスを使用して、同じ記憶場所に対する多数の記憶アクセスの間で整合性を執行する種々の方法および装置で実現して、プロセッサ上で稼働するプログラムに提示される結果的なデータ・シーケンスが記憶場所に存在するデータ・シーケンスと整合するようにすることができる（すなわち、命令がそれらの元のプログラム順序で実行されたならば、プログラムがそのようなデータ・シーケンスを認識することができたであろう）。より具体的には、方法は、処理装置を制御して、論理的に先行するインオーダ・ロード動作がデータを読み出すところのアドレスを、論理的なプログラム順序ではそのインオーダ命令に続くアウトオブオーダ・ロード命令によって投機アクセスされたアドレスと比較させる干渉試験からなる。アドレスどうしがマッチするならば、アウトオブオーダ・ロード命令に論理的に先行するインオーダ・ロード命令は、投機実行されるアウトオブオーダ・ロード命令に提供されたものと同じデータを提示される。

【００５３】データをアウトオブロード命令からインオーダ・ロード命令にバイパスすることは、書き込み直列化されたプログラム実行を維持しながら、アウトオブオーダ・ロード命令およびその従属命令を再実行する必要を回避させる。

【００５４】本発明のバイパス機構を使用する効果を示すために、先の例を参照する。左側の命令ストリームが、二重発行ユニプロセッサの場合に生成されるような順序変更されたプログラム・ストリームを示す。本発明のバイパス機構は、データを干渉するアウトオブオーダ・ロード（「*」によって示す）から左側に示すプログラムのインオーダ・ロード（「＋」によって示す）にバスするために使用される。この順序変更されたプログラム（左側）の実行は、以下の例の右側で指定されるインオーダ・プログラムの実行に等しく、したがって、正しい書き込み直列実行である。

【００５５】

インオーダ実行に等しい

第二のプロセッサが、ｓおよびｔへのロードによって参照される記憶にデータｄ１を書き込む。

バイパスで順序変更されたコード

```
        mul     r10,r1,4        mul     r10,r1,4
*       load    r5,(r9)   add             r11,r10,5
        add     r11,r10,5       add             r12,r11,r2
                                load            r4,(r12)
```

```
                                        add             r6,r4,4
                                        load        r5,(r9)
```

第二のプロセッサがデータをｄ２に変形する。

```
        add             r12,r11,2   add                 r7,r5,8
        add             r7,r5,8
+       load    r4,(r12)
        add             r6,r4,4
```

【００５６】本発明の干渉試験およびバイパス論理は、プロセッサ中のハードウェア資源、プロセッサによって実行される命令またはハイブリッド・ハードウェア／ソフトウェア・スキームによって実現することができる。ソフトウェア実現形態の場合、干渉試験は、新たな一つの命令からなることもできるし、既存の命令（たとえば、ロード命令および比較・条件付き移動命令またはロード命令、比較命令、分岐命令および移動命令）からなることもできる。干渉試験の具体的な実現形態は、記憶動作の順序変更が静的に実行されるのか動的に実行されるのかに依存する。アウトオブオーダ・ロード動作に提供されるアドレスおよびデータ値は、正規データ値として主記憶中に格納することもできるし、命令ストリーム中の命令から明示的にアクセス可能であってもよいし、そうでなくてもよい、プロセッサに埋め込まれたハードウェア・テーブル構造中に格納することもできる。提案されるスキームを、ロード／ストア干渉を検出するためのスキームおよび干渉を検出したとき、いくつかのロード動作を再発行するためのスキームと合わせて、異なる記憶整合性モデルを実現することができる。このような場合、これらのスキームに要する一つ以上の資源をロード結果バイパス資源と合わせて、汎化アウトオブオーダ実行スキームにおけるハードウェアの複雑さを軽減することができる。

【００５７】まず図１を参照すると、記憶動作以外の動作の順序変更を静的または動的に実行する場合の、従来技術による命令のスケジューリングを示す流れ図が示されている。順序変更は、そのような順序変更が曖昧さを被らない場合に限り、図１の方法にしたがって実行される。

【００５８】ステップ１０１で、次の命令を選択し、それが有利であり、プログラムの正しい動作に影響しないならば、命令シーケンス中のその現在位置からより早い位置に移動する（すなわち、命令を実行のためにスケジュールする）。すなわち、選択される命令が、その命令が移動されるときに追い越すいずれの命令にも従属しないならば、その命令をより早い位置に移動する。

【００５９】ステップ１０３で、スケジュールすべき命令が残っているかどうかを決定する。残っているならば、プロセスはステップ１０１に戻る。残っていないならば、プロセスは終了する。

【００６０】図２は、本発明の実施態様による、元の命令シーケンスからの動作を順序変更するプロセスを示す流れ図である。特に、図２は、本発明にしたがって、スケジューリング技術を拡張して曖昧さ解消を扱う方法を示す。この一般的なスキームは、曖昧さ解消を実現するすべての技術（すなわち静的または動的）に応用可能であることがわかる。ここで示す特定のアルゴリズムは、スケジュールされる各命令は、その前にスケジュールされたすべての命令の後で実行される（すなわち、選択順序が実行順序を反映する）という仮定に基づく。しかし、当業者は、他のスケジューリング手法とで使用するためにこのアルゴリズムの詳細を変更する方法を理解するであろう。

【００６１】ステップ２０１で、次の命令を選択し、命令シーケン中のその現在位置からより早い位置に移動する（すなわち、命令を実行のためにスケジュールする）。

【００６２】ステップ２０２で、選択命令がロード（または読み取りアクセスのための記憶場所を参照するかもしれない他の命令）であったかどうかを決定する。ロードであったならば、ステップ２０２で、同じ記憶場所を曖昧に参照するかもしれない非選択命令（たとえば、読み取りアクセスのための記憶場所を参照するかもしれない非選択ロード命令または他の命令）が以前にその選択されたロード命令を越えて移動されたかどうかをさらに決定する。ステップ２０２で下される第二の決定はまた、選択された（すなわち現在の）命令が、一つ以上のアウトオブオーダ命令に対してインオーダ命令であるかどうかとして特徴付けることもできる。ステップ２０２で下される二つの決定の結果のいずれかがＮＯであるならば、プロセスはステップ２０４に進む。しかし、ステップ２０２で、選択命令がロード命令であり、さらに、同じ記憶場所を曖昧に参照するかもしれない非選択命令が、選択されたロード命令を越えて移動されたと決定されるならば、プロセスはステップ２０３に進む。

【００６３】ステップ２０３で、曖昧さ解消を扱う動作を実行する。これらは、選択されたロード命令に対してアウトオブオーダであるロード命令の間に干渉（すなわち、選択されたロード命令を越えて移動されたロード命令による干渉）が存在するかどうかを決定するための試験を加えること、およびそのような干渉の場合に実行されるバイパス・シーケンスを準備することを含むが、これらに限定はされない。このバイパス・シーケンスは、選択されたインオーダ・ロード命令が先のアウトオブオーダ・ロード・シーケンスと同じ値を返すことを保証す

ることにより、正しい時間順の（書き込み直列の）イベントを執行しなければならない。バイパス・シーケンスは、多数の（アウトオブオーダ）ロード動作が選択されたロード命令を越えて移動されたことがあるかどうかを多数のバイパス条件で検査する。バイパス・シーケンス中に実行される検査ステップは、ハードウェア手段、ロード命令中に符号化されたフラグもしくは他の手段、さらなる命令またはさらなる命令シーケンスによって実現することができる。

【００６４】多数の検査を実行するならば、最終的な結果が参照される記憶場所に対して書き込み直列になることを保証するように実行しなければならない。これを保証するため、多数の干渉するロード命令が、選択されたインオーダ・ロード命令に対してアウトオブオーダで実行されるならば、その値を、干渉するもっとも早い論理的に後続するアウトオブオーダ・ロード命令のレコードからバイパスしなければならない。好ましくは、これは二つの方法のいずれかを使用して実行される。第一の方法では、干渉およびバイパス試験を論理的なプログラム順序で実行し、ひとたび干渉が検出されたならば、特定の不可分性単位のためのすべてのさらなる試験およびバイパス（ロード動作で多数の不可分性単位が参照されるならば）をスキップする。

【００６５】第二の方法では、逆の論理的プログラム順序で干渉試験およびバイパスを実行する。いずれの方法も書き込み直列な挙動を保証することが理解されよう。さらに、当業者が、二つの方法の教示に基づき、本発明の本質または範囲を逸することなく、それに対して種々の変更および変形を加えることができることが理解されよう。

【００６６】ステップ２０４で、選択命令がロードであったかどうかを決定し、ロードであったならば、選択命令が以前に同じ記憶場所を曖昧に参照するかもしれない別の非選択命令（たとえば、読み取りアクセスのために記憶場所を参照するかもしれない非選択ロード命令または他の命令）を越えて移動されたかどうかをさらに決定する。ステップ２０４で下される二つの決定はまた、選択命令が、一つ以上のインオーダ命令に対するアウトオブオーダ・ロード命令であるかどうかとして特徴付けることができる。ステップ２０４で下される二つの決定のいずれかの結果がＮＯであるならば、プロセスはステップ２０６に進む。しかし、ステップ２０４で、選択命令がロード命令であり、さらに、選択命令が以前に同じ記憶場所を曖昧に参照するかもしれない別の非選択命令を越えて移動されたと決定されるならば、プロセスはステップ２０５に進む。

【００６７】ステップ２０５で、アウトオブオーダ・ロード命令が移動されたときに追い越したすべてのインオーダ・ロード命令による将来の参照のためにこのアウトオブオーダ・ロード命令の結果を記録するための機構を加える。そのような動作を記録するための機構は、ハードウェア手段、ロード命令中に符号化された「フラグ」（命令ビット）もしくは他の手段、さらなる命令またはさらなる命令シーケンスからなることができる。必要ならば、そのようなレコードを削除する（また、そのような情報を格納するために要した資源を取り戻す）ためのさらなるコードを、移動されたロード動作があった元のプログラム位置に挿入してもよい。

【００６８】ステップ２０６で、スケジュールすべき命令が残っているかどうかを決定する。残っているならば、プロセスはステップ２０１に戻る。残っていないならば、プロセスは終了する。

【００６９】図３は、図２の順序変更された動作を実行するプロセスを示す流れ図である。具体的には、図３は、図２のバイパス・ステップおよび記録ステップをより詳細に説明する。

【００７０】ステップ２５１で、アウトオブロード命令を実行し、それにより、データを読み取り、そのデータを、アウトオブオーダ・ロード命令で指定されたレジスタに格納する。ステップ２５３で、アドレスおよびロード命令によってロードされた値を将来の参照のために記録する。ステップ２５５で、場合によっては、ステップ２５３でロードされた値に従属する命令を含む他の命令を実行してもよい。

【００７１】ステップ２５７で、ステップ２５１で実行された先のアウトオブオーダ・ロード命令が移動されたときに追い越したインオーダ命令を実行する。ステップ２５９で、検査を実行して、二つのロード命令のアドレスどうしが干渉するかどうかを決定する。アドレスどうしが干渉するならば、プロセスはステップ２６１に進む。しかし、アドレスどうしが干渉しないならば、プロセスはステップ２６３に進む。

【００７２】ステップ２６１で、先のアウトオブオーダ・ロード動作からのロード値をバイパスして、アウトオブオーダ・ロードに提供（によってロード）されたデータ値が次にインオーダ・ロードに提供（によってロード）されるようにする。

【００７３】ステップ２６３で、ロード命令のいずれか（または両方）に従属する命令または独立した命令を含む他の命令を実行してもよい。ステップ２６５で、アウトオブオーダで実行された元のロード命令の位置で、ステップ２５３で割り当てたロード・レコードの項目を割り当て解除する。

【００７４】いくつかのロード命令が別のロード命令を越えて移動されるならば、インオーダ・ロード命令に対してアウトオブオーダに移動されたロード命令ごとにステップ２５９および２６１を繰り返すことにより、いくつかの検査を必要に応じて実行することがわかる。さらに、ある命令が、いくつかのアウトオブオーダ命令に対してはインオーダ命令であり、かつ、いくつかのインオ

ータ命令に対してはアウトオブオーダ命令であるならば、記録ステップ２５３の前にバイパス・ステップ２６１を実行することが好ましい。

【００７５】上記プロセスの最適化として、干渉が検出されない場合にだけインオーダ記憶アクセスが実行されるようにステップ２５７を順序変更することができる。

【００７６】本手法は、静的スキームおよび動的スキームで使用することができる。これら実現形態は、二つの動作が互いに対してインオーダであるのかアウトオブオーダであるのか、また、参照がどのように生成され、参照されるのかを識別するために異なる手法を使用することができる。

【００７７】二つのロード動作の間でインオーダ／アウトオブオーダ関係を識別する一つの方法は、アウトオブオーダ・ロード動作に識別子を割り当てたのち、その識別子をインオーダ・ロード動作の干渉およびバイパス試験によって使用して、プログラムの流れの中でそのようなインオーダ動作に対してアウトオブオーダである動作を識別することによる方法である。このような識別子は、コンパイラによって静的に割り当てることもできるし、発行ハードウェアによって動的に生成することもできる。

【００７８】本発明を記憶語の点で説明したが、これはまた、ビットおよびバイトでアドレス指定可能な記憶をはじめとする種々の組織形態を有する記憶構造にも適用できることが理解されよう。そのうえ、記憶動作は、記憶語が部分的に重複している（すなわち、二つの記憶語によって参照されるビットのサブセットだけがこれらの語に共通である）ことを指定してもよい。このような場合、これら共通のビットだけがバイパスされる。さらに、記憶システムは、記憶アクセスのための不可分性要件を指定することができる。そのような場合、適当な不可分性単位（通常はバイト、語またはキャッシュ・ライン）が、インオーダ・ロード命令にバイパスされるものとしてアウトオブオーダ・ロード命令によって記録される。さらには、本発明の本質および範囲を保持しながらも、異なる記憶モデルとで使用するための他の変形を本発明に加えることもできる。

【００７９】干渉試験およびデータ・バイパスのために静的スケジューリングおよび命令を使用する実施態様

静的スケジューリングおよび命令を使用する本発明の干渉試験に可能な実現形態を説明するため、以下、干渉試験に望まれる機能性を語レベルで実現する例を提供する。しかし、ビット、バイト、２分語または他のレベルで記憶読み取り動作を実現するプロセッサを、以下に記載する方法の使用に適合させることもできることが理解されよう。そこで、アドレス指定可能な単位ごとに種々の項目からのバイパスの可能性に関して、別々にアドレス指定可能な単位ごとに独立して比較を実施する。システムはさらに、バイト、２分語、語キャッシュ・ラインまたはより高位のレベルで不可分性を保証することができる。そのような不可分性が保持されなければならないならば、本発明は、アウトオブオーダ命令によって生成されるレコード中の周囲の不可分性単位を記録するように適合させることができる。

【００８０】本明細書で提供する例は、以下の表記を使用する。

・ＲＴ、ＲＡは汎用レジスタを表し、

・（ＲＴ）、（ＲＡ）は、汎用レジスタＲＴ、ＲＡそれぞれの内容であり、

・Ｄ、ＢＤは、それぞれデータ変位および分岐変位であり、

・ＣＩＡ、ＮＩＡは、それぞれ、現在実行中の命令のアドレスおよび次に実行される命令のアドレスである。

・別段記さない限り、疑似コードは、ＩＳＯ　Ｃ規格によるＣプログラミング言語に指定されるものに類似した演算子を使用する。たとえば、[]は索引を指定し、.(dot)はレコード・フィールドを指定する。

【００８１】静的スケジューリングの一つの例示的な実現形態は、コンパイラによって各アウトオブオーダ・ロード動作に割り当てられるロードタグ識別子を使用する。ロードタグ識別子と一つのアウトオブオーダ・ロード動作とは、そのロード動作のアウトオブオーダ発行点とそのような動作の論理的インオーダ点との関係のように、一意に対応する。

【００８２】ロードタグ識別子は、各アウトオブオーダ動作命令語の中に格納され、図３でステップ２５３によって生成されるレコードの中に記録される。このようなアウトオブオーダ・ロード命令は、その命令フォーマット中に一つのさらなるフィールドを含み、ロードタグ識別子４を有するアウトオブオーダ・ロード命令を以下load_4　RT,D(RA)と表記する。

【００８３】この動作は、ロードを実行することに加えて、ロード結果（不可分性を保証するためにこのアクセスを含む最大の不可分性単位）、ロードタグ識別子およびアドレスをテーブルに記録する。

【００８４】dealloc_4と指定される命令が、図３のステップ２６５に示すように、ロードタグｉｄ４を有するテーブル項目を割り当て解除する。dealloc_n動作は、図３のプロセスで任意の（なくてもよい）ステップである。したがって、具体的に実現形態に依存して、dealloc_n動作をコードから省いてもよく、テーブル項目が後の割り当て要求によって自動的に取って代わられてもよい。

【００８５】一つの効率的な実現形態は、このロードタグ識別子を、アウトオブオーダ・ロード結果に関する情報を記録するために使用されるテーブルの中で実際の索引として使用することからなる。そして「アドレス干渉および条件付きロード・レジスタを検査せよ」命令（定義は以下に記す）を使用して、バイパス条件の干渉試験

および実際のバイパス・ステップを実現することができる。この命令は、アドレスを比較する相手であり、アドレス干渉の場合に結果をバイパスされるアウトオブオーダ・ロードを指定する。

【００８６】アドレス（先に実行されたアウトオブオーダ・ロード命令がデータをロードしたところ）が、現在指定されているアドレス（インオーダ・ロード命令の）とマッチする（または、部分語動作の場合には、重複する）場合、アドレス干渉が検出される。好ましくは、アドレス干渉は、以下の命令にしたがって検出される。

【００８７】アドレス干渉および条件付きロード・レジスタを検査せよ

```
chkld_n RT,D(RA)
EA<;=(RA)+D
if(entry[n].address==EA)
        ;               bypass if addresses interfere
        RT<;=entry[n].value
```

【００８８】ＥＡは、インオーダ命令によって参照される有効記憶アドレスである。計算された有効アドレスが、chkld命令（ｎ）によって指定された項目に記録されている、先に実行されたアウトオブオーダ・ロードの記憶域とマッチする、または重複するならば、対応する値はバイパスされる。

【００８９】chkld命令は実質的にステップ２５９および２６１を一つの命令で実現する。

【００９０】chkld命令を実際のロード命令と合わせて、ステップ２５７、２５９および２６１を一つの検査付き動作ロードに合わせられることがわかる。

【００９１】

```
load_chkld_n RT,D(RA)

EA<;=(RA)+D
if(entry[n].address==EA)
        ;               bypass if addresses interfere

        RT<;=entry[n].value
else
        RT<;=MEM(EA,4)
```

【００９２】あるいはまた、二つの記憶アドレスのアドレス干渉ならびに重複の場合にはそれらの対応するサイズおよび分岐を試験して、ソフトウェアベースの干渉およびバイパス試験を加速する一つの試験アドレス干渉および分岐命令だけを使用することが望ましいかもしれない。

【００９３】本発明の干渉試験およびバイパス動作（「アドレス干渉および条件付きロード・レジスタを検査せよ」命令（chkld））の静的生成およびその実行を説明するため、以下、第一のロード命令の下の第二のロード命令および第二のロード命令に従属するいくつかの算術命令を含む、左側の欄に示す元のコードを考えてみよう。この例では、命令の名前の後の最初のレジスタが標的レジスタであり、残りのレジスタはオペランドである。

【００９４】

元のコード		インオーダ・ロード命令を越えて移動されるロード命令		
...		load_3	r5,10(r4)	
...		...		
...		...		
load	r3,20(r)	load	r3,20(r2)	
		chkld_3		r3,20(r2)
...		...		
load	r5,10(r4)	dealloc_3		
add	r6,r5,20	add	r6,r5,20	
sub	r7,r6,r7	sub	r7,r6,r7	
$next: ...		$next: ...		

【００９５】右側の欄に示すように、第二のロード命令が第一のロード命令を越えて移動される。インオーダ・ロード命令が「アドレス干渉および条件付きロード・レジスタを検査せよ」命令（chkld）によって増補され、元の第二のロード命令（アウトオブオーダで移動されたもの）が、右側の欄に示すように、バイパス・レコードを割り当て解除するための命令によって取って代わられる。元のロード命令に続く算術命令もまた、以下に示すように順序変更することができる。

【００９６】

元のコード		インオーダ・ロード命令を越えて移動される動作	
...			load_3 r5,10(r4)
...			add r6,r5,20
...			sub r7,r6,r7
...			...
load	r3,20(r)	load	r3,20(r2)
			chkld_3 r3,20(r2)
...			...
load	r5,10(r4)	dealloc_3	
add	r6,r5,20	$next: ...	
sub	r7,r6,r7		
$next: ...			

【００９７】この場合、右側の欄に示すように、ロード動作に従属する命令もまた、インオーダ命令を越えて移動される。

【００９８】以下に示すように、ロードと検査を組み合わせた命令（load_chkld_n）を使用して、命令ロードおよびchkldを一つの動作に畳む。

【００９９】

元のコード		インオーダ・ロード命令を越えて移動される動作	
...			load_3 r5,10(r4)
...			add r6,r5,20
...			sub r7,r6,r7
...			...
load	r3,20(r)	load_chkld_3	r3,20(r2)
			...
load	r5,10(r4)	dealloc_3	
add	r6,r5,20	$next: ...	
sub	r7,r6,r7		
$next: ...			

【０１００】図４は、記憶動作の静的順序変更ならびに干渉試験およびデータ・バイパス・シーケンスのソフトウェア実現形態をサポートする従来のコンピュータ処理システムの機能ブロック図である。このシステムは、記憶サブシステム３０１、データ・キャッシュ３０２、命令キャッシュ３０４および処理装置３００からなる。処理装置３００は、命令待ち行列３０３、１個以上のロード装置３０５（１個だけ示す）、整数、論理および浮動小数点演算を実行するためのいくつかの実行装置３０７、分岐装置３０９ならびにレジスタ・ファイル３１１を含む。命令は、分岐装置３０９の制御の下、命令キャッシュ３０４から（または、命令キャッシュ３０４にない場合には記憶サブシステム３０１から）取り出され、命令待ち行列３０３に配置される。命令は、実行されるため、命令待ち行列３０３からロード装置３０５、実行装置３０７および分岐装置３０９にディスパッチされる。これらの装置３０５、３０７、３０９は、レジスタ・ファイル３１１と対話して、命令によって使用されるオペランドにアクセスし、命令の実行によって出される結果を保存する。レジスタ・ファイル３１１は通常、汎用レジスタ（ＧＰＲ）、浮動小数点レジスタ（ＦＰＲ）および条件レジスタ（ＣＲ）を含む。ロード装置３０５はまた、データ・キャッシュ３０２および記憶サブシステム３０１と対話して、実行装置３０７および／または分岐装置３０９によって実行される命令によって使用されるデータをロードし、実行装置によって生成される結果を格納する。

【０１０１】図５は、本発明の実施態様にしたがって図４のシステムによるアウトオブオーダ・ロード動作の実行をサポートするために使用されるハードウェア資源を示すブロック図である。より具体的には、図５は、図３に示す少なくとも一つのロード装置３０５の詳細ブロック図である。少なくとも一つのロード装置３０５は、制御論理４０１、アドレス生成論理４０２、ロードオーダ・バッファ４０３、比較論理４０４およびバイパス用データ選択機構４０５を含む。

【０１０２】第一の作動モードでは、通常のロード命令を処理するとき、アドレス生成論理４０２が、データを読み出すところの記憶アドレスを生成する。ロード命令の具体的なフォーマットに依存して、記憶アドレスは、ロード命令で指定されるレジスタ・ファイル３１１のレジスタから読み出されるオペランドに基づいて生成することもできるし、レジスタ・ファイル３１１のレジスタから読み出されるオペランドと、ロード命令で指定される変位情報とに基づいて生成することもできる。そして、アドレス生成論理４０２は、生成した記憶アドレス

をデータ・キャッシュ３０２に転送する。要求されたデータは、データ・キャッシュ３０２または記憶サブシステム３０１（データ・キャッシュ３０２にない場合）から読み出されたのち、制御論理４０１によって制御される選択機構４０５によってレジスタ・ファイル３１１中の宛先レジスタに戻され、書き込まれる。

【０１０３】第二の作動モードでは、アウトオブオーダ・ロード命令を処理するとき、アドレス生成論理４０２は、データを読み出すところの記憶アドレスを生成する。ロード命令の具体的なフォーマットに依存して、記憶アドレスは、ロード命令で指定されるレジスタ・ファイル３１１のレジスタから読み出されるオペランドに基づいて生成することもできるし、レジスタ・ファイル３１１のレジスタから読み出されるオペランドと、ロード命令で指定される変位情報とに基づいて生成することもできる。そして、アドレス生成論理４０２は、生成した記憶アドレスをデータ・キャッシュ３０２に転送する。要求されたデータは、データ・キャッシュ３０２または記憶サブシステム３０１（データ・キャッシュ３０２にない場合）から読み出されたのち、制御論理４０１によって制御される選択機構４０５によってレジスタ・ファイル３１１中の宛先レジスタに戻され、書き込まれる。同時に、制御論理４０１が、ロードオーダ・バッファ４０３に指図して、データがロードされたところのアドレス、データ型（場合による）、要求されたデータを含む下層アーキテクチャによって不可分性であることを保証された最大単位およびロードタグ識別子（図５では、索引としてテーブル（すなわちロードオーダ・バッファ４０３）中に暗示的に格納されている）からなるレコードを格納させる。

【０１０４】第三の作動モードでは、本発明の「アドレス干渉および条件付きロード・レジスタを検査せよ」命令（干渉試験およびバイパス）を処理するとき、アドレス生成論理４０２は、データを読み出すところの記憶アドレスを生成する。「アドレス干渉および条件付きロード・レジスタを検査せよ」命令の具体的なフォーマットに依存して、記憶アドレスは、「アドレス干渉および条件付きロード・レジスタを検査せよ」命令で指定されるレジスタ・ファイル３１１のレジスタから読み出されるオペランドに基づいて生成することもできるし、レジスタ・ファイル３１１のレジスタから読み出されるオペランドと、「アドレス干渉および条件付きロード・レジスタを検査せよ」命令で指定された変位情報とに基づいて生成することもできる。

【０１０５】制御論理４０１は、「アドレス干渉および条件付きロード・レジスタを検査せよ」命令で指定されたロードタグ識別子に対応するレコードにアクセスし、含まれた記憶アドレスを比較論理４０４に供給する。さらに、アドレス生成論理４０２が、「アドレス干渉および条件付きロード・レジスタを検査せよ」命令の結果として生成された記憶アドレスを比較論理４０４に供給すると、この比較論理が両アドレスを比較する。アドレスどうしがマッチする、または記憶重複を示すならば、比較論理４０４は、それをデータ選択機構４０５に指示する。アドレスのマッチまたは重複の場合、ロードオーダ・バッファ４０３中でアクセスされた項目のデータ・フィールドに格納されたデータがライトバックのためにレジスタ・ファイル３１１に供給される。すなわち、はじめにアウトオブオーダ・ロード命令に供給されたデータ（ロードオーダ・バッファ４０３に格納されていたもの）が、今や、インオーダ・ロード命令に提供される。そうでなければ、インオーダ・ロード命令のためにレジスタ・ファイル３１１中に現在格納されている値は変化しない（すなわち、アウトオブオーダ・ロードとインオーダ・ロードとの間で干渉が検出されなかった）。選択機構４０５は、制御論理４０１の制御の下、インオーダ・ロードのためにレジスタ・ファイル３１１に書き込むべき、ロードオーダ・バッファ４０３からバイパス項目によって供給されるビットのサブセットのみを選択することができる。これは、記憶システムの不可分性単位を超える、干渉するアウトオブオーダおよびインオーダ命令の部分的な重複の場合に適切であろう。

【０１０６】第四の作動モードでは、命令dealloc_nを実施して、ロードオーダ・バッファ４０３からの項目を割り当て解除する。この命令は任意（なくてもよい）であり、したがって、望むならば省いてもよい。

【０１０７】上述した実施態様は、アウトオブオーダ・ロード命令が移動されるときに追い越すロード命令ごとに一つの余計な命令（すなわち、「アドレス干渉および条件付きロード・レジスタを検査せよ」命令）を取り入れ、一つ以上の他のロード命令に対してアウトオブオーダに移動されるロード命令ごとに一つの命令（すなわち、dealloc_n命令）を取り入れる。これは、元のプログラムに対してオーバヘッドであるが、コストでは、一つ以上の命令を使用してバイパス条件を検出したのち、干渉が検出されたときに回復シーケンスを実行する従来機構との比較で優位である。従来機構によるこのシーケンスは、命令キャッシュへの悪影響および回復コードの存在がレジスタ割り当て機構に課す制限に加えて、コードを格納するための空間と、コードを実行するための時間の両方を要する。本発明に関しては、ロード動作をアウトオブオーダで移動する決定がプログラム生成時に下されるため、オーバヘッドの影響は、移動を実行することで期待される利点とで比較考量することができる。

【０１０８】もう一つの代替態様では、アレイ（たとえばロードオーダ・バッファ４０３）中の先のアウトオブオーダ・ロード動作に関する情報を命令比較シーケンス（または前記の一つの記憶重複命令）によって使用して記憶重複を試験したのち、アドレス重複の場合に、さらなる命令シーケンスによって使用して、アレイに格納さ

れた値を、条件付きで、インオーダ・ロード命令によって指定された標的レジスタに移動する。さらなる命令は、条件付き移動命令もしくは条件付き移動命令が続く分岐命令またはアウトオブオーダ・ロード命令によって生成されたレコードに格納された値をインオーダ・ロード命令によって指定された標的レジスタ中に移動する効果を有する他の命令シーケンスを含むことができる。

【０１０９】さらに別の代替態様では、本発明は、ブロック内で連続的に実行される命令に関して、命令を非循環コード・ブロック（たとえば、基本ブロックまたはスーパブロック）中で厳密に降順の命令アドレスでスケジュールするために使用される。そのような場合、アウトオブオーダ・ロード命令によって指定されたロードタグ識別子が、元のロード命令のその論理的プログラム順序におけるプログラム・カウンタ位置である。図７は、本発明の実施態様にしたがって図４または６のシステムによるアウトオブオーダ・ロード動作の実行をサポートするために使用されるハードウェア資源を示すブロック図である。具体的には、図７は、本発明の実施態様の、図４に示すロード装置３０５の実現形態の詳細ブロック図とみなすことができる。ロード装置３０５は、制御論理６０１、アドレス生成論理６０３、ロードオーダ・テーブル６０５、干渉試験および優先順位エンコーダ６０７、第一のバイパス用データ選択機構６０９および第二のバイパス用データ選択機構６１１を含む。

【０１１０】投機ロード命令が実行されるとき、ロード装置３０５が、データがロードされるところの記憶アドレスと、記憶から受け取られたデータ値と、投機ロード命令の論理的プログラム位置（命令番号）とを連想記憶テーブル６０５に記録する。論理的プログラム位置は、多くの方法で指定することができる。たとえば、命令中に符号化されたさらなる命令フィールド、特別な命令前置子または一つ以上のさらなるコンピュータ命令を使用することができる。

【０１１１】インオーダ命令が実行されるとき、ロード装置３０５は、アドレス生成論理６０３によって生成された記憶ロード・アドレスに対してＣＡＭアクセスを実行し、そのようなアドレスが以前にそれ自体よりも高いプログラム位置（命令番号としてテーブル６０５に格納されている）を有するアウトオブオーダ・ロード動作によってアクセスされたことがあるかどうかを検査し（アウトオブオーダ・ロードがこのテーブルにアクセスする場合には、そのロード動作の論理的プログラム位置を使用する）、「干渉試験および優先順位エンコーダ」６０７中の同じ記憶場所を参照する。アドレス生成装置６０３によって生成された記憶ロード・アドレスが以前にアウトオブオーダ・ロード動作によってアクセスされたことがあるならば、以前に処理されたアウトオブオーダ・ロードと現在のロード動作とが干渉することが検証され、データ選択機構６１１中の干渉する項目と対応するデータからのバイパスが可能にされる。ある特定の記憶アドレスに対応する多数の項目がテーブル６０５中に存在し、そのうち、二つ以上が干渉していると思われることがある。多数の項目が一つのロード動作に干渉するならば、干渉試験および優先順位エンコーダ６０７は、優先順位符号化スキームを使用して、アウトオブオーダ・ロード命令によってはじめに指定され、テーブル６０５に格納されている最下位の論理的プログラム・アドレス（命令番号としてテーブル６０５に格納されている）を有する干渉項目からデータが提供されることを保証する。

【０１１２】プログラム・カウンタがアウトオブオーダ・ロード命令の元の論理的プログラム順序の値に達すると、テーブル項目は制御論理６０１によって割り当て解除される。

【０１１３】この機構は、プログラム空間をハードウェア複雑さで相殺する。実証することができるとおり、そのような実施態様を可能にするためにさらなる命令は要らない。静的スケジューリングは、テーブル６０５の不十分な記憶容量のせいでそのテーブルに格納することができないロード命令がアウトオブオーダにスケジュールされることがないように実現されるべきである。連想記憶装置がテーブル６０５に好ましい実現形態であるが、当業者が、テーブル６０５と同様な機能を果たし、本発明の本質および範囲を維持する他の記憶構造を考えられることを理解されたい。

【０１１４】干渉試験およびデータ・バイパスのために動的スケジューリングおよびハードウェア資源を使用する実施態様

命令を動的にスケジュールすることができるプロセッサ（アウトオブオーダ発行プロセッサ）の従来の実現形態は、以下の機構を含む。

【０１１５】１．命令をアウトオブオーダで発行する機構であって、命令間の従属性を検出し、命令によって使用されるレジスタをリネームし、命令によって使用される資源の可使性を検出する能力を有するもの。

２．プロセッサのアウトオブオーダ状態を維持するための機構であって、命令がアウトオブオーダで実行された場合の命令の影響を反映するもの。

３．命令をプログラム順序で撤収し、同時に、撤収される命令の効果でインオーダ状態を更新するための機構。

４．命令をプログラム順序で撤収するが、インオーダ状態を更新せず（撤収される命令の影響を実質的に打ち消す）、撤収される命令から出発してプログラムのインオーダ状態を回復する（アウトオブオーダ状態で存在するすべての影響をを打ち消すことを暗示する）ための機構。

【０１１６】上記リストのうち機構３は、撤収される命令の影響が正しいとき、その命令を撤収するために使用される。あるいはまた、撤収される命令の実行または何

らかの外部的イベントから生じる異常な状態が存在するときには、機構4が使用される。

【0117】図6は、記憶動作の動的順序変更ならびに干渉試験およびデータ・バイパス・シーケンスのハードウェアベースの実現形態をサポートする従来のコンピュータ処理システム（たとえばスーパスケーラ・プロセッサを含む）の機能ブロック図である。すなわち、図6のシステムは、上述した機構を使用する命令の順序変更をサポートするのに必要であるハードウェア資源を含むが、インオーダ動作よりも前のアウトオブオーダ・ロード動作の実行をサポートするのに必要であるハードウェア資源を含まない。システムは、記憶サブシステム501、データ・キャッシュ502、命令キャッシュ504および処理装置500からなる。処理装置500は、命令待ち行列503、ロードおよびストア動作を実行するためのいくつかの記憶装置（MU）505、整数、論理および浮動小数点演算を実行するためのいくつかの機能装置（FU）507、分岐装置（BU）509、レジスタ・ファイル511、レジスタ・マップ・テーブル520、フリーレジスタ待ち行列522、ディスパッチ・テーブル524、撤収待ち行列526およびインオーダ・マップ・テーブル528を含む。この例示的な組織は、M. Moudgill、K.PingaliおよびS. Vassiliadisによる論文「Register Renaming and DynamicSpeculation: An Alternative Approach」Proceedings of the 26th Annual International Symposium on Microarchitecture, pp. 202-13（1993年12月）に記載されたものに基づく。

【0118】図6に示すプロセッサでは、命令は、分岐装置509の制御の下、命令キャッシュ504（または命令キャッシュ504にない場合には記憶サブシステム501）から取り出され、命令待ち行列503に配置されたのち、命令待ち行列503からディスパッチされる。オペランドを指定するために命令によって使用されるレジスタ名は、アーキテクチャ・レジスタ名から物理レジスタへの現在のマッピングを指定する、レジスタ・マップ・テーブル520の内容にしたがってリネームされる。結果の宛先を指定するために命令によって使用されるアーキテクチャ・レジスタ名は、現在プロセッサによって使用されていない物理レジスタの名前を含むフリーレジスタ待ち行列522から抽出される物理レジスタを割り当てられる。レジスタ・マップ・テーブル520は、物理レジスタを、命令によって指定されるアーキテクチャ宛先レジスタ名に割り当てることによって更新される。すべてのレジスタをリネームされた命令は、ディスパッチ・テーブル524に配置される。命令はまた、それらのアドレスならびにそれらの物理およびアーキテクチャ・レジスタ名を含めて、撤収待ち行列526の中にプログラム順序で配置される。命令は、そのような命令によって使用されるすべての資源が使用可能になったとき（物理レジスタが予期されるオペランドを割り当てられ、機能装置が未使用であるとき）、ディスパッチ・テーブル524からディスパッチされる。命令によって使用されるオペランドは、通常は汎用レジスタ（GPR）、浮動小数点レジスタ（FPR）および条件レジスタ（CR）を含むレジスタ・ファイル511から読み出される。命令は、対応する記憶装置505、機能装置507または分岐装置509の中で潜在的にアウトオブオーダで実行される。実行が完了すると、命令の結果がレジスタ・ファイル511に配置される。実行を完了する命令によってセットされる物理レジスタを待つ、ディスパッチ・テーブル524中の命令が通知される。撤収待ち行列526は、実行を完了する命令を、それらの命令が何らかの例外を提起したかどうかを含め、通知される。完了した命令は、撤収待ち行列526からプログラム順序で（行列の先頭から）削除される。撤収時に、命令によって例外が提起されなかったならば、アーキテクチャ・レジスタ名がレジスタ・ファイル511中の、撤収される命令の結果を含む物理レジスタを指し示すよう、インオーダ・マップ・テーブル528が更新される。インオーダ・マップ・テーブル528から先のレジスタ名はフリーレジスタ待ち行列522に戻される。

【0119】他方、命令が例外を提起したならば、プログラム制御は、撤収待ち行列526から撤収される命令のアドレスにセットされる。そのうえ、撤収待ち行列526はクリアされ（フラッシュされ）、それにより、すべての未撤収命令が打ち消される。さらに、レジスタ・マップ・テーブル520がインオーダ・マップ・テーブル528の内容にセットされ、インオーダ・マップ・テーブル528中にないレジスタがフリーレジスタ待ち行列522に加えられる。

【0120】上記部品に加えて、スーパスケーラ・プロセッサは、他の部品、たとえば分岐の結果を予測するための分岐履歴テーブルを含むことができる。

【0121】本発明によると、先行するロード命令に対するロード命令の順序変更をサポートする従来のスーパスケーラ・プロセッサ（図6に示す）を以下によって増補することができる。

【0122】1．先行するロード命令に対してアウトオブオーダで発行されるロード命令を印すための機構。

2．命令が取り出されるとき、それらの命令に番号を付け、命令が命令ストリーム中で早く起こったか遅れて起こったかを決定するための機構。命令が別の命令に対して早く起こったか遅れて起こったかを決定するための代替機構を用いてもよい。

3．アウトオブオーダで実行されたロード動作に関する情報、たとえばプログラム順序におけるそれらのアドレス、それらのアクセスのアドレスおよびロードされたデータを含む最大保証不可分性単位のための読み出されたデータ値を格納するための機構。

４．ロード命令が一つ以上のアウトオブオーダ・ロード命令に対してインオーダで実行されるときに干渉試験を実行し、多数の命令がロード命令に干渉するときに優先順位符号化を実行するための機構。
５．干渉するロード動作に対応するデータをバイパスするための機構。
６．アウトオブオーダ状態がプログラム順序で撤収待ち行列５２６からレジスタ・ファイル５１１に撤収される点でステップ３で生成されたレコードを削除するための機構。
【０１２３】本発明によって提供される機構は、以下に記すように、図６に示す従来のアウトオブオーダ・プロセッサで利用できる機構と併せて使用される。各命令は、命令待ち行列５０３に加わるとき、命令番号を付される。ロード命令は、先行するロード命令よりも先にディスパッチテーブル５２４からディスパッチすることができる。以下、そのようなロード命令を「アウトオブオーダ・ロード動作」と呼ぶ。そのような場合、ロード命令に対応する撤収待ち行列５２６中の項目は、アウトオブオーダ・ロードと印される。
【０１２４】ディスパッチ・テーブル５２４から記憶装置５０５へのアウトオブオーダ・ロード動作の実行のためのディスパッチの検出は、２個のカウンタ、「取り出し済みロード・カウンタ」および「ディスパッチ済みロード・カウンタ」によって達成することが好ましい。取り出し済みロード・カウンタは、ロード動作がディスパッチ・テーブル５２４に加えられるごとに増加する。ディスパッチ済みロード・カウンタは、ロード動作が実行のために記憶装置５０５に送られるごとに増加する。ロード命令がディスパッチ・テーブル５２４に加えられるとき、取り出し済みロード・カウンタの現在の内容がロード命令に付加される。ロード命令が実行のためにディスパッチ・テーブル５２４から記憶装置５０５にディスパッチされるとき、ディスパッチテーブル５２４中のロード命令に付加された値がその時点でのディスパッチ済みロード・カウンタの内容と異なるならば、そのロード命令はアウトオブオーダ・ロード命令と識別される。２個のカウンタ値の違いが、ロード命令がアウトオブオーダで発行されるときに追い越されるロード動作の正確な数に対応することがわかる。アウトオブオーダ・ロード命令は、ロードオーダ・テーブル６０５に項目を加えるための空間が利用できる場合にのみ、記憶装置５０５にディスパッチされる。
【０１２５】ロードオーダ・テーブル６０５は、すべての記憶装置５０５によって同時にアクセスされる一つのテーブルである。すなわち、処理を加速するために多数の物理コピーを維持することはできるが、論理コピーは一つしか維持されない。多数の物理コピーが使用されるならば、それら多数のコピーの論理内容は常にすべての記憶装置５０５の同じ状態を反映しなければならないことがわかる。
【０１２６】ロード動作が発行されるごとに、実行されている命令の命令番号および命令が投機実行されるのかどうかが記憶装置５０５に送られる。
【０１２７】図７は、本発明の実施態様にしたがって、図６のシステムによってアウトオブオーダ動作の実行をサポートするために使用されるハードウェア資源を示すブロック図である。具体的には、図７は、図６に示す記憶装置５０５の詳細ブロック図である。記憶装置５０５は、制御論理６０１、アドレス生成論理６０３、ロードオーダ・テーブル６０５、干渉試験および優先順位エンコーダ６０７、第一のバイパス用データ選択機構６０９および第二のバイパス用データ選択機構６１１を含む。干渉試験および優先順位符号化論理６０７によって操作される第一のデータ選択機構６０９は、ロードオーダ・テーブル６０５から論理的にもっとも早い干渉するアウトオブオーダ・ロードのデータ値を選択するために使用される。第二のデータ選択機構６１１は、データ・キャッシュ５０２（またはキャッシ・ミスの場合には、記憶サブシステム５０１）によって返されるデータまたはロードオーダ・テーブル６０５からのデータを選択するために使用される。第二のデータ選択機構６１１によって選択されるデータは、干渉試験および優先順位符号化論理６０７によって干渉が検出されたかどうかに依存して、レジスタ・ファイル５１１に対する将来のコミットに使用される。第一および第二のデータ選択機構６０９、６１１によって実行される機能は、一つのデータ選択機構に統合してもよいことが理解されよう。ロードオーダ・テーブル６０５の好ましい実現形態は連想記憶装置（ＣＡＭ）である。しかし、当業者は、実質的に同じ効果を有する他の記憶構造をそれに代えて用いてもよいことを理解するであろう。
【０１２８】ロード動作を受けると、アドレス生成装置６０３によってアドレスが生成され、データ・キャッシュ５０２にディスパッチされる。そして、データ・キャッシュ５０２がアクセスされてデータを提供する。データがデータ・キャッシュ５０２中に見つからないならば、記憶サブシステム５０１がアクセスされてデータを提供する。同時に、ロードオーダ・テーブル６０５が、制御論理６０１により、アドレス生成装置６０３によって生成されたデータ・アドレスでアクセスされる。干渉試験および優先順位エンコーダ６０７が干渉を求めて検査する。ロードオーダ・テーブル６０５中に、データ・キャッシュ５０２の現在のアクセスと同じアドレスを参照する項目（または、現在のアドレスのアドレスを含む不可分性単位のそれ）が存在し、そのレコードが、現在実行中のロード命令よりも後の命令番号をそれに対応して有する（すなわち、記録されたロード命令が、現在処理中のロード命令に対してアウトオブオーダである）ならば、干渉が検出される。干渉が検出されるならば、干

渉試験および優先順位エンコーダ装置６０７は、データ選択機構６１１に指図して、情報をロードオーダ・テーブル６０５からレジスタ・ファイル５１１にパスさせるか、データ・キャッシュ５０２から他の方法でパスさせる。最適な実施態様では、バイパス条件に遭遇するならば、データ・キャッシュ５０２または記憶サブシステム５０１へのアクセスを抑制することができる。

【０１２９】多数のマッチに遭遇するならば、干渉試験および優先順位エンコーダ６０７は、データ選択機構６０９に指図して、もっとも早い命令番号（現在の命令の番号よりも後）を有するテーブル項目を、データ選択機構６１１のためのバイパス・データ・ソースとして選択させる。

【０１３０】実行されたロード動作がアウトオブオーダと印されているならば、レジスタ・ファイル５１１に供給されるデータ値は、命令の論理的プログラム・アドレスおよびデータ記憶アドレスとともに、ロードオーダ・テーブル６０５の中の項目に記録される。ロード命令は、いくつかのアウトオブオーダ命令に対してインオーダであり、かつ他のインオーダ・ロード命令に対してアウトオブオーダであることができることがわかる。したがって、バイパスおよび記録論理の両部分を一つの命令によって執行してもよい。

【０１３１】アウトオブオーダ・ロード動作を撤収待ち行列５２６から撤収すると、命令または制御信号が記憶装置５０５に送られて、撤収された命令に対応する項目が削除される（たとえば一意の命令番号を指定することにより）。

【０１３２】分岐予測ミスに遭遇し、一つ以上の投機実行されたアウトオブオーダ・ロード命令が撤収バッファからフラッシュされるならば、本発明にしたがって、記憶装置５０５のバイパス・テーブル構造中の適切な項目が同時に削除される。

【０１３３】命令番号は、このスキームでは、入力されたプログラムの逐次（インオーダ）実行における命令の順序を確立するために使用される。命令番号は、所与の時間にプロセッサによって処理することができるすべての動作を一意に識別することができなければならず、所与の時間にプロセッサによって同時に処理される二つの命令の関係を確立するために使用することができなければならない。たとえば、命令番号は、プロセッサ中で所与の時点でインフライトである（処理される）ことができる数の少なくとも２倍の数ｎの命令を符号化するのに十分な広さの一つの番号からなることができる。そして、モジュロ算術が、現在の命令番号に先行するｎ－１の範囲にある番号を有する命令が現在の命令に論理的に先行し、現在の命令番号に続くｎ－１の範囲にある番号を有する命令が現在の命令に論理的に後続することをを確立することができる。

【０１３４】代替の番号付けスキームは、命令アドレスおよび分岐がとられるごとのさらなる分岐前置子を使用することからなることができる。分岐前置子は、いかなる数の形態、たとえば分岐テーブルへの参照のモジュロ・カウンタを取ることができる。命令番号付けの代替実現形態は、命令ストリームに対して順序付けを課すハードウェア・テーブルを参照することができる。これは、既存のハードウェア要素、たとえば撤収待ち行列５２６または命令を順序付けする専用の目的のために実現されたハードウェア機構に基づくことができる。さらに別の代替形態は、前記スキームにしたがって、たとえば取り出し済みロード・カウンタ値を使用してロード動作に番号付けすることにより、ロード命令をカウントするだけであってもよい。

【０１３５】本発明の代替実施態様は、命令番号に基づいて項目を格納する代わりに、アクセスされる記憶アドレスに基づいてロードオーダ・バイパス・テーブル項目を生成することができる。この実施態様は、干渉試験および優先順位エンコーダ６０７における優先順位符号化の要件を除くが、互いに対してインオーダであるが、第三のロード命令に対してはアウトオブオーダである命令の間でさらなるバイパス条件を生じさせる。そのような実行がなおも書き込み直列化されるが、そのような挙動は、所与の記憶場所に対する読み取りに関してプロセッサによって返される値が他のプロセッサによって提示される値ストリームと同期化される速度を落とすため、望ましくないかもしれないことがわかる。

【０１３６】本発明のもう一つの実施態様は、命令を動的に生成し、命令待ち行列５０３に挿入することができる。この実施態様は、干渉試験およびデータ・バイパスの静的実施態様に関して上記したものに類似した命令スケジュールを動的に生成する。したがって、プロセッサによって実行される命令は、静的実施態様と動的実施態様とで類似しており、唯一の違いは、スケジュールが、動的実施態様の場合にはプログラム実行時にプロセッサによって生成され、静的実施態様の場合にはコンパイラによって生成されることにある。

【０１３７】添付図面を参照しながら例示的な実施態様を本明細書に記載したが、本システムおよび方法はこれらの正確な実施態様に限定されず、本発明の範囲または本質を逸することなく、他にも種々の変更および変形が当業者によって加えられることが理解されよう。そのようなすべての変更および変形は、請求の範囲によって定義される本発明の範囲に含まれるものとされる。

【０１３８】まとめとして、本発明の構成に関して以下の事項を開示する。

（１）アウトオブオーダ実行を実現するコンピュータ処理システムで命令を実行のためにスケジュールする方法であって、次の命令を選択し、命令シーケンス中のその現在位置からより早い位置に移動するステップと、選択命令が読み取りアクセスのために前記記憶場所を参照で

きるかどうかを決定するステップと、選択命令が読み取りアクセスのために前記記憶場所を参照できる場合、読み取りアクセスのために前記記憶場所を曖昧に参照できる非選択命令が選択命令を越えてあらかじめ移動されたかどうかを決定するステップと、非選択命令が選択命令を越えてあらかじめ移動されており、非選択命令がデータを読み取りアクセスした記憶場所のアドレスが、選択命令によってデータが読み取りアクセスされる記憶場所のアドレスと同じである場合、選択命令の実行中に実行され、選択命令に非選択命令によってあらかじめ読み取りアクセスされたデータをパスするバイパス・シーケンスを設けるステップと、選択命令が読み取りアクセスのために記憶場所を参照できる場合、選択命令が以前に非選択命令を越えて移動されたかどうかを決定するステップと、非選択命令による将来の参照のために、選択命令の記録を格納するための機構を追加するステップと、を含むことを特徴とする方法。

（２）選択命令が非選択命令を越えてあらかじめ移動された場合、前記追加するステップを実行する上記（１）記載の方法。

（３）非選択命令が読み取りアクセスのための前記記憶場所を曖昧参照できる場合、前記追加するステップを実行する上記（１）記載の方法。

（４）選択すべき命令が残っているかどうかを決定するステップと、選択するべき命令が残っている場合、前記選択するステップに戻るステップと、をさらに含む上記（１）記載の方法。

（５）異なる命令を越えてあらかじめ移動された第一の命令に、異なる命令を識別するロードタグ識別子を割り当てるステップをさらに含む上記（１）記載の方法。

（６）非選択命令が以前に選択命令を越えて移動されたかどうかを決定する前記ステップが、ロードタグ識別子が非選択命令に割り当てられたことがあるかどうかを決定するステップと、ロードタグ識別子が選択命令を識別するかどうかを決定するステップと、を含む上記（５）記載の方法。

（７）選択命令が以前に非選択命令を越えて移動されたかどうかを決定する前記ステップが、ロードタグ識別子が選択命令に割り当てられたことがあるかどうかを決定するステップと、ロードタグ識別子が非選択命令を識別するかどうかを決定するステップと、を含む上記（５）記載の方法。

（８）ロードタグ識別子をアウトオブオーダ・ロード命令語とともに格納するステップをさらに含む上記（５）記載の方法。

（９）アウトオブオーダ実行を実現するコンピュータ処理システムで命令を実行のためにスケジュールする方法であって、次の命令を選択し、命令シーケンス中のその現在位置からより早い位置に移動するステップと、選択命令が読み取りアクセスのために記憶場所を参照するかもしれないかどうかを決定するステップと、選択命令が読み取りアクセスのために記憶場所を参照するかもしれない場合、選択命令が、読み取りアクセスのために記憶場所を参照するかもしれない非選択命令に対してインオーダ命令であるかどうかを決定するステップと、選択命令が、非選択命令に対してインオーダ命令であり、非選択命令がデータを読み取りアクセスしたところの記憶場所のアドレスが、選択命令がデータを読み取りアクセスするところの記憶場所のアドレスと同じである場合、前記選択命令の実行中に実行される、非選択命令によって以前に読み取りアクセスされたデータを選択命令にパスするバイパス・シーケンスを設けるステップと、選択命令が読み取りアクセスのために記憶場所を参照するかもしれない場合、前記選択命令が非選択命令に対してアウトオブオーダ命令であるかどうかを決定するステップと、前記選択命令が前記非選択命令に対してアウトオブオーダ命令である場合、前記非選択命令による将来の参照のために、選択命令のレコードを格納するための機構を追加するステップと、を含むことを特徴とする方法。

（１０）選択命令が非選択命令に対してアウトオブオーダ命令である場合、前記追加ステップを実行する上記（９）記載の方法。

（１１）非選択命令が読み取りアクセスのために記憶場所を曖昧に参照するかもしれない場合、前記追加ステップを実行する上記（９）記載の方法。

（１２）レコードが、選択命令がデータを読み取りアクセスしたところの記憶アドレスと、データに対応する値とを含む上記（９）記載の方法。

（１３）各アウトオブオーダ命令に識別子を割り当てて、アウトオブオーダ命令およびアウトオブオーダ命令に対してインオーダである命令を識別するステップをさらに含む上記（９）記載の方法。

（１４）識別子をアウトオブオーダ命令語の中に格納する上記（１３）記載の方法。

（１５）識別子を、それに割り当てられたアウトオブオーダ命令のレコードの中に記録する上記（１３）記載の方法。

（１６）アウトオブオーダ命令に割り当てられた識別子で識別されるインオーダ命令が実行されており、干渉しないものであると決定されている場合、アウトオブオーダ命令のレコードを破壊するための手段を追加するステップをさらに含む上記（１３）記載の方法。

（１７）アウトオブオーダ命令のレコードが後の割り当て要求によって自動的に取って代わられるように設ける上記（９）記載の方法。

（１８）レコード中に格納されたデータ単位が、不可分性を保証する可能な最大データ単位である上記（９）記載の方法。

（１９）多数のアウトオブオーダ命令がインオーダ命令と同じアドレスを参照する場合、最下位の論理プログラ

ム・アドレスを有するデータがインオーダ・ロード命令にパスされることを保証するために優先順位符号化を実現するステップをさらに含む上記（９）記載の方法。

（２０）命令シーケンスが少なくとも一つの処理装置によって実行され、多数の読み取りアクセスがアウトオブオーダ・ロード命令および少なくとも一つの他のロード命令によって起動されるコンピュータ処理システム中の記憶場所からの多数の読み取りアクセスの間に得られるデータ値の間の整合性を執行する方法であって、アウトオブオーダ・ロード命令を実行して少なくとも一つの処理装置を制御して、アウトオブオーダ・ロード命令によって識別される記憶場所から第一のデータを少なくとも読み取りアクセスさせるステップと、少なくとも一つの他のロード命令が使用するための、第一のデータがロードされたところの記憶場所のアドレスと、第一のデータに対応する値とを含む、アウトオブオーダ・ロード命令のレコードを生成するステップと、少なくとも一つの他のロード命令を実行し、少なくとも一つの他のロード命令の実行中に少なくとも一つの処理装置を制御して、アウトオブオーダ命令が第一のデータをロードしたところの記憶場所のアドレスが、少なくとも一つの他のロード命令がデータをロードするところのアドレスと重複する、または同じであるかどうかを決定するステップと、アドレスが同じである、または重複する場合、それぞれ第一のデータまたはその一部をレコードから少なくとも一つの他のロード命令にパスするステップと、を実行させるステップと、を含むことを特徴とする方法。

（２１）処理装置が命令シーケンス中のアウトオブオーダ・ロード命令の元の位置に達した場合、レコードを破壊するステップをさらに含む上記（２０）記載の方法。

（２２）命令シーケンス中のアウトオブオーダ・ロード命令の元の位置でレコードを削除するための手段を追加するステップをさらに含む上記（２０）記載の方法。

（２３）各アウトオブオーダ命令に識別子を割り当てて、アウトオブオーダ命令およびアウトオブオーダ命令に対してインオーダである命令を識別するステップをさらに含む上記（２０）記載の方法。

（２４）アウトオブオーダ命令を実行する前記ステップが、少なくとも一つの処理装置を制御して、第一のデータをアウトオブオーダ命令によって識別される第一の標的レジスタ中に配置させるステップをさらに含む上記（２０）記載の方法。

（２５）前記決定ステップおよびパスするステップが、少なくとも一つの処理装置によって実行される一つの命令を含む上記（２０）記載の方法。

（２６）命令シーケンスが少なくとも一つの処理装置によって実行され、多数の読み取りアクセスが少なくとも一つのアウトオブオーダ・ロード命令および一つのインオーダ・ロード命令によって起動されるコンピュータ処理システムの記憶サブシステム中の場所からの多数の読み取りアクセスの間に得られるデータ値の間の整合性を執行するための装置であって、データが読み取りアクセスされたところの記憶サブシステム中の場所のアドレスと、データに対応する値とを含む、少なくとも一つのアウトオブオーダ・ロード命令のレコードを格納するための格納手段と、インオーダ命令を処理するとき、少なくとも一つのアウトオブオーダ・ロード命令がデータを読み取りアクセスしたところの記憶サブシステム中の場所のアドレスと、インオーダ・ロード命令がデータを読み取りアクセスするところの記憶サブシステム中の場所のアドレスとを比較するための比較論理と、前記格納手段に指図してレコードを格納させ、少なくとも一つのアウトオブオーダ命令のレコードにアクセスしてアドレスを前記比較論理に供給するための制御論理と、前記比較論理によって比較されたアドレスがそれぞれ等しい、または重複する場合、前記制御論理の制御の下で、前記格納手段に格納された値またはその一部をインオーダ命令に供給するためのデータ選択機構と、を含むことを特徴とする装置。

（２７）前記格納手段が、ＣＡＭ構造またはレコードを格納するために予め指定された前記記憶サブシステム中の区域である上記（２６）記載の装置。

（２８）前記レコードが、前記アウトオブオーダ・ロード命令に対応するインオーダ・ロード命令を識別するための識別子をさらに含む上記（２６）記載の方法。

（２９）前記識別子で識別されるように前記アウトオブオーダ命令に対応する前記インオーダ命令が実行されており、干渉しないものであると決定されている場合、記格納手段に格納されたレコードを削除するための手段をさらに含む上記（２８）記載の装置。

（３０）少なくとも一つのアウトオブオーダ・ロード命令およびインオーダ・ロード命令を処理するとき、データを読み取りアクセスするところの記憶サブシステム中の場所のアドレスを生成するためのアドレス生成論理をさらに含む上記（２６）記載の装置。

（３１）インオーダ・ロード命令がデータを読み取りアクセスするところの記憶中の場所のアドレスからデータを読み取りアクセスしたことがある多数のアウトオブオーダ・ロード命令が存在する場合、命令シーケンス中で最下位の元の位置を有するデータを検出するための優先順位符号化手段をさらに含む上記（２６）記載の装置。

（３２）命令シーケンスが少なくとも一つの処理装置によって実行され、多数の読み取りアクセスが少なくとも一つの先に実行されたロード命令および現在実行中のロード命令によって起動されるコンピュータ処理システムの記憶サブシステム中の場所からの多数の読み取りアクセスの間に得られるデータ値の間の整合性を執行するための装置であって、データが読み取りアクセスされたところの記憶サブシステム中の場所のアドレスと、データとを含む、少なくとも一つの先に実行された命令のレコ

ードを格納するための格納手段と、現在実行中の命令を処理するとき、少なくとも一つの先に実行された命令がデータを読み取りアクセスしたところの場所のアドレスと、現在実行中の命令がデータを読み取りアクセスするところの場所のアドレスとを比較し、少なくとも一つの先に実行された命令が、現在実行中の命令よりも後の命令番号をそれに対応して有するかどうかを決定するための干渉試験および優先順位エンコーダと、前記格納手段に指図してレコードを格納させ、少なくとも一つの先に実行された命令のレコードにアクセスしてアドレスを前記干渉試験および優先順位エンコーダに供給するための制御論理と、前記前記干渉試験および優先順位エンコーダによって比較されたアドレスが等しい、または重複し、少なくとも一つの先に実行された命令が、より遅い命令番号をそれに対応して有する場合、前記干渉試験および優先順位符号化論理の制御の下で、前記格納手段に格納されたデータまたはその一部を現在実行中の命令に供給するためのデータ選択機構と、を含むことを特徴とする装置。

（３３）前記干渉試験および優先順位エンコーダによって比較されるアドレスが等しくなく、重複もしない場合、前記データ選択機構が、記憶サブシステム中に格納されたデータを現在実行中の命令に供給する上記（３２）記載の装置。

（３４）少なくとも一つの先に実行された命令が二つ以上存在する場合、前記データ選択機構が、論理的にもっとも早い干渉する先に実行された命令に対応するデータを格納手段から現在実行中の命令に供給する上記（３２）記載の装置。

（３５）現在実行中の命令がデータを読み取りアクセスするところの記憶中の場所のアドレスからデータを読み取りアクセスしたことがある少なくとも一つの先に実行された命令が二つ以上存在する場合、前記干渉試験および優先順位エンコーダが、命令シーケンス中で最下位の元の位置を有するデータを検出する上記（３２）記載の装置。

（３６）命令番号が、処理装置によって所与の期間に処理することができる数の少なくとも２倍の数ｎの命令を符号化するのに十分に広い上記（３２）記載の装置。

（３７）少なくとも一つの先に実行された命令の命令番号が、現在実行中の命令の命令番号に続くｎ－１の範囲にある場合、少なくとも一つの先に実行された命令が現在実行中の命令よりも後の命令番号を有する上記（３６）記載の装置。

【図面の簡単な説明】

【図１】従来技術による、記憶動作以外の動作の順序変更を実行する場合の命令のスケジューリングを示す流れ図である。

【図２】本発明の実施態様による、元の命令シーケンスから動作を順序変更する方法を示す流れ図である。

【図３】図２の順序変更された動作を実行する方法を示す流れ図である。

【図４】記憶動作の静的順序変更ならびに干渉試験およびデータ・バイパス・シーケンスのソフトウェアベースの実現形態をサポートする従来のコンピュータ処理システムの機能ブロック図である。

【図５】本発明の実施態様にしたがって、図３のシステムによるアウトオブオーダ・ロード動作の実行をサポートするために使用されるハードウェア資源を示すブロック図である。

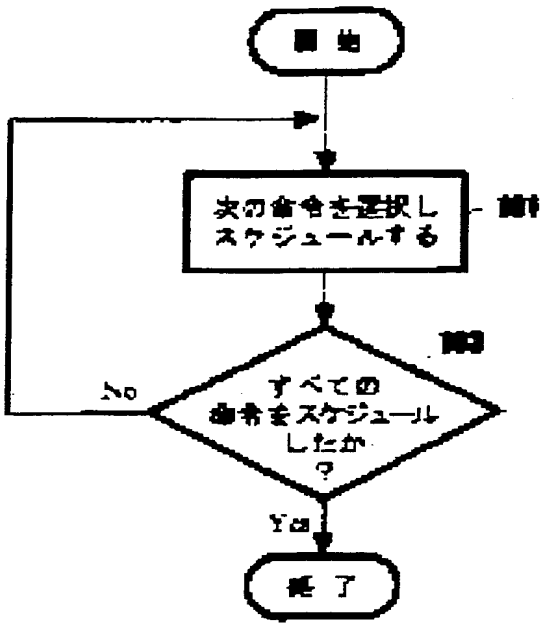
【図６】記憶動作の動的順序変更ならびに干渉試験およびデータ・バイパス・シーケンスのハードウェアベースの実現形態をサポートする従来のコンピュータ処理システムの機能ブロック図である。

【図７】本発明の実施態様にしたがって、図４または６のシステムによるアウトオブオーダ・ロード動作の実行をサポートするために使用されるハードウェア資源を示すブロック図である。

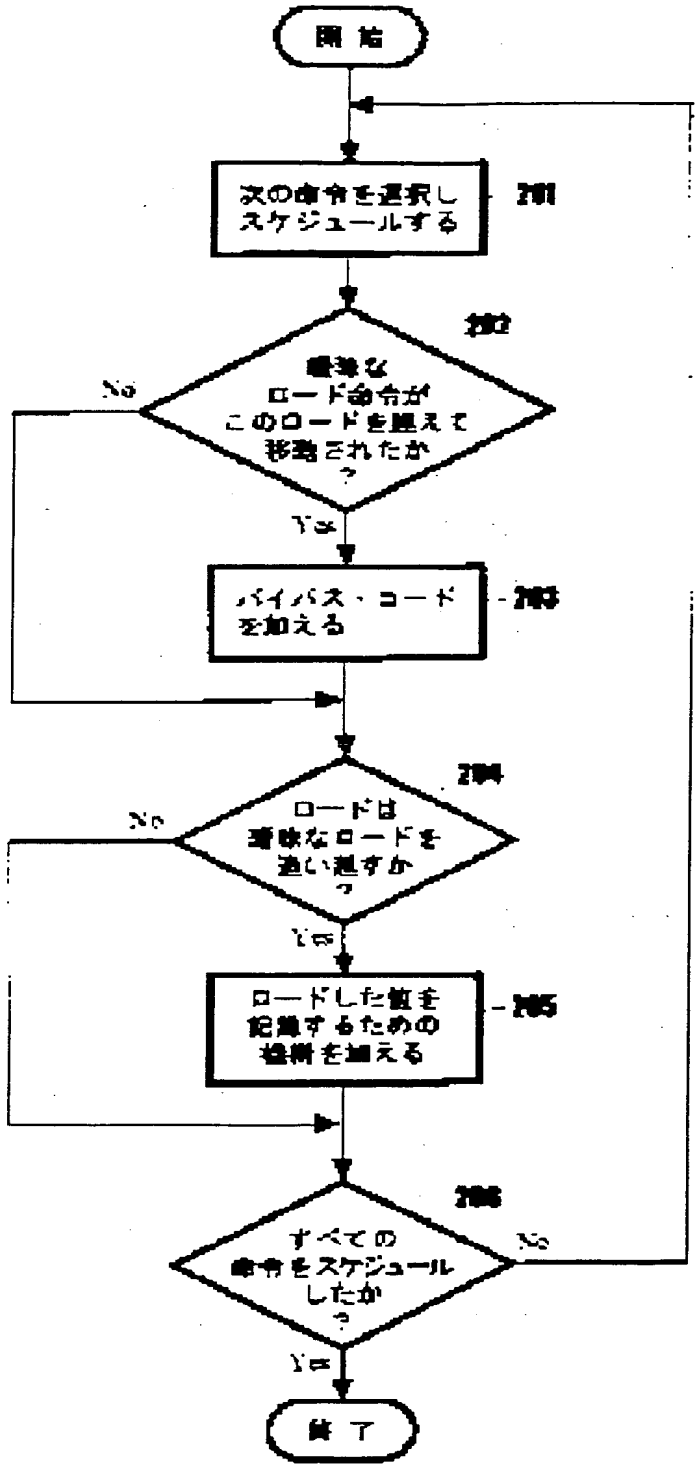
【符号の説明】

３０１、５０１　記憶サブシステム
３０２、５０２　データ・キャッシュ
３０３、５０３　命令待ち行列
３０４、５０４　命令キャッシュ
３０５　ロード装置
３０７　実行装置
３０９、５０９　分岐装置
３１１　レジスタ・ファイル
４０１、６０１　制御論理
４０２、６０３　アドレス生成論理
４０３、６０５　ロードオーダ・テーブル
４０４　比較論理
４０５、６０９、６１１　バイパス用データ選択機構
６０７　干渉試験および優先順位エンコーダ

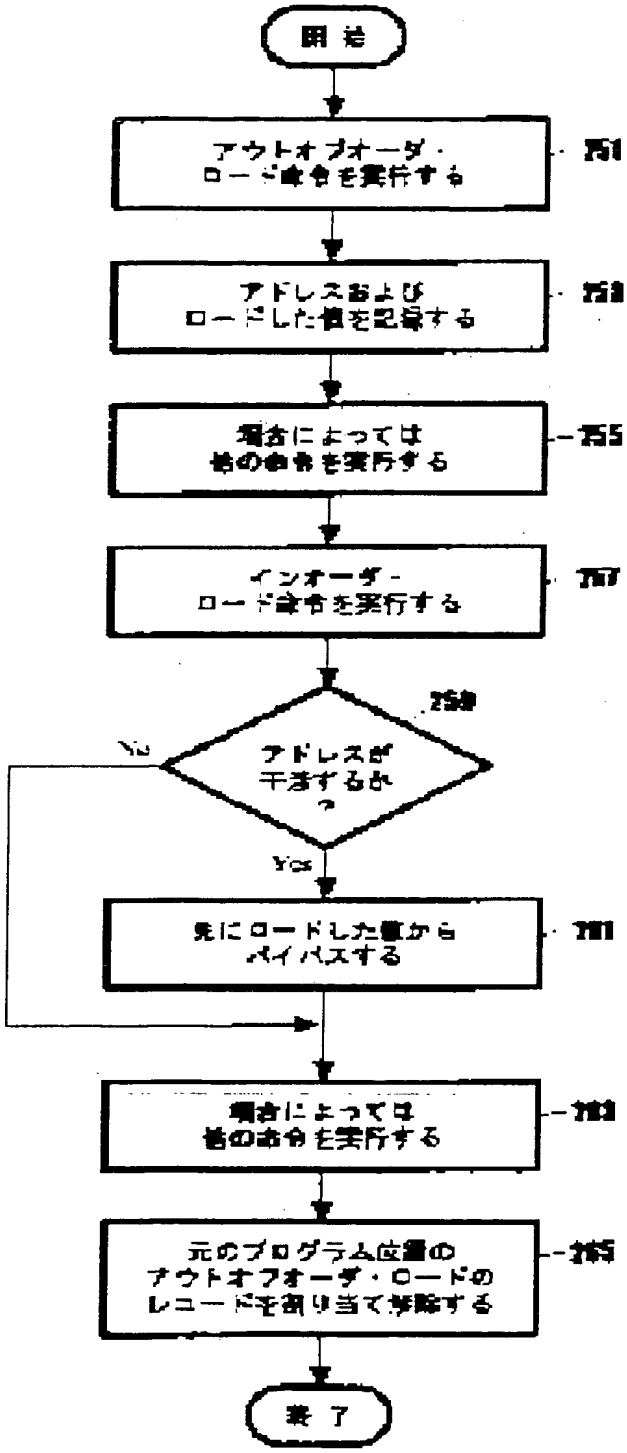
【図1】



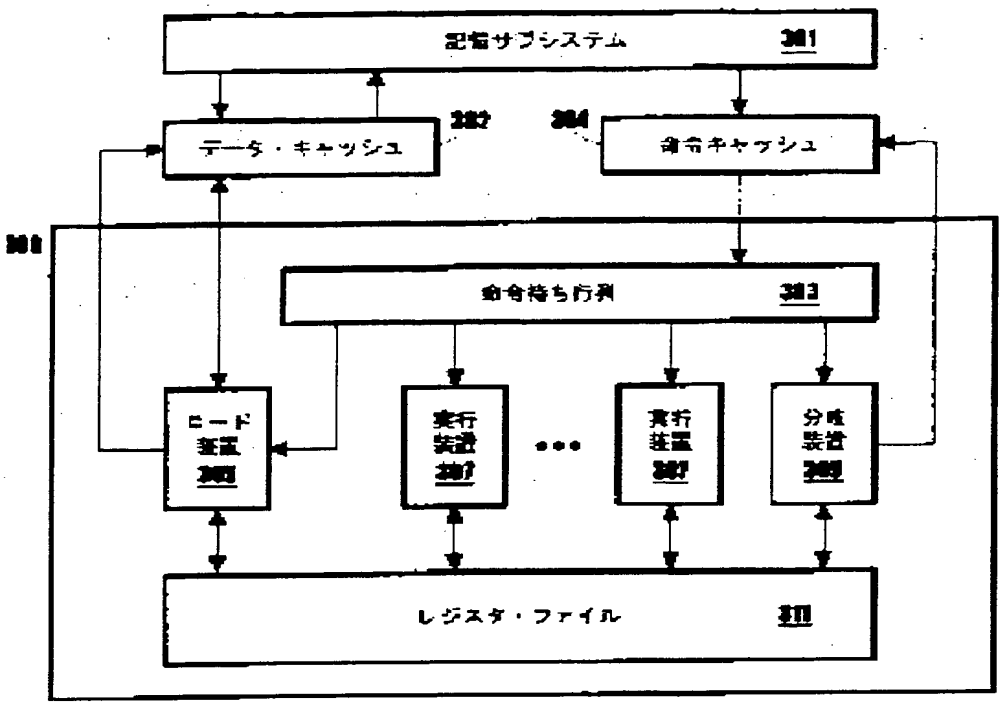
【図2】



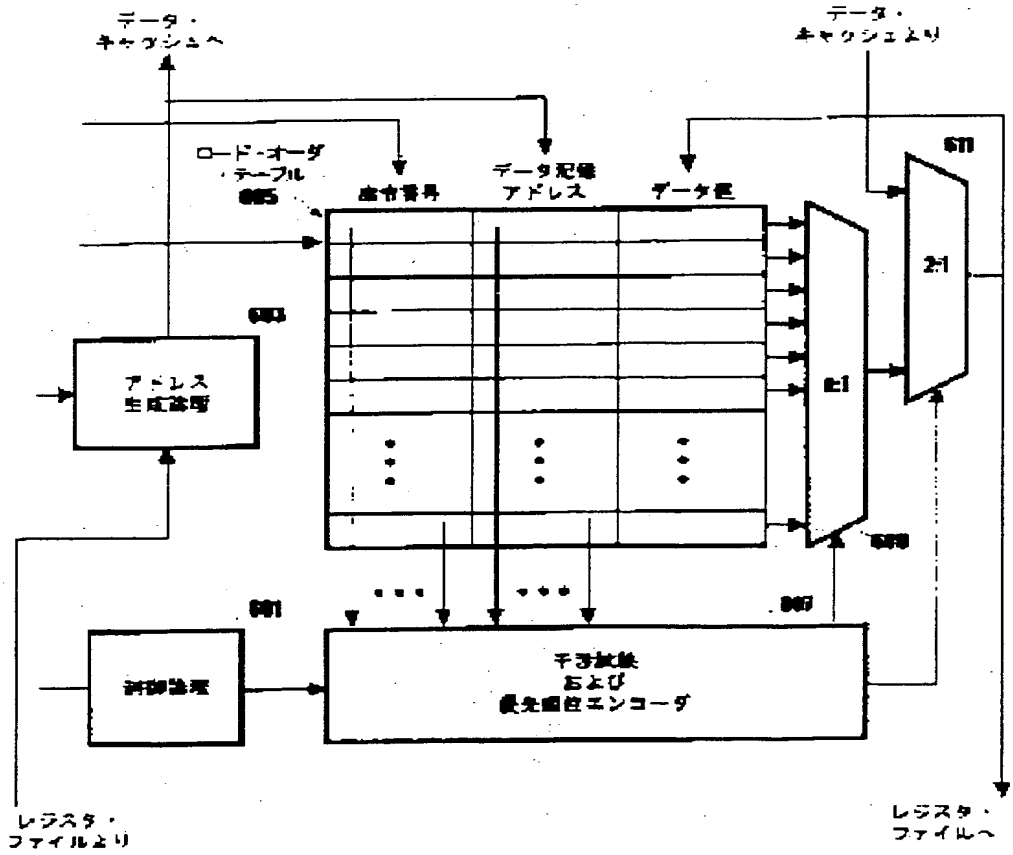
【図3】



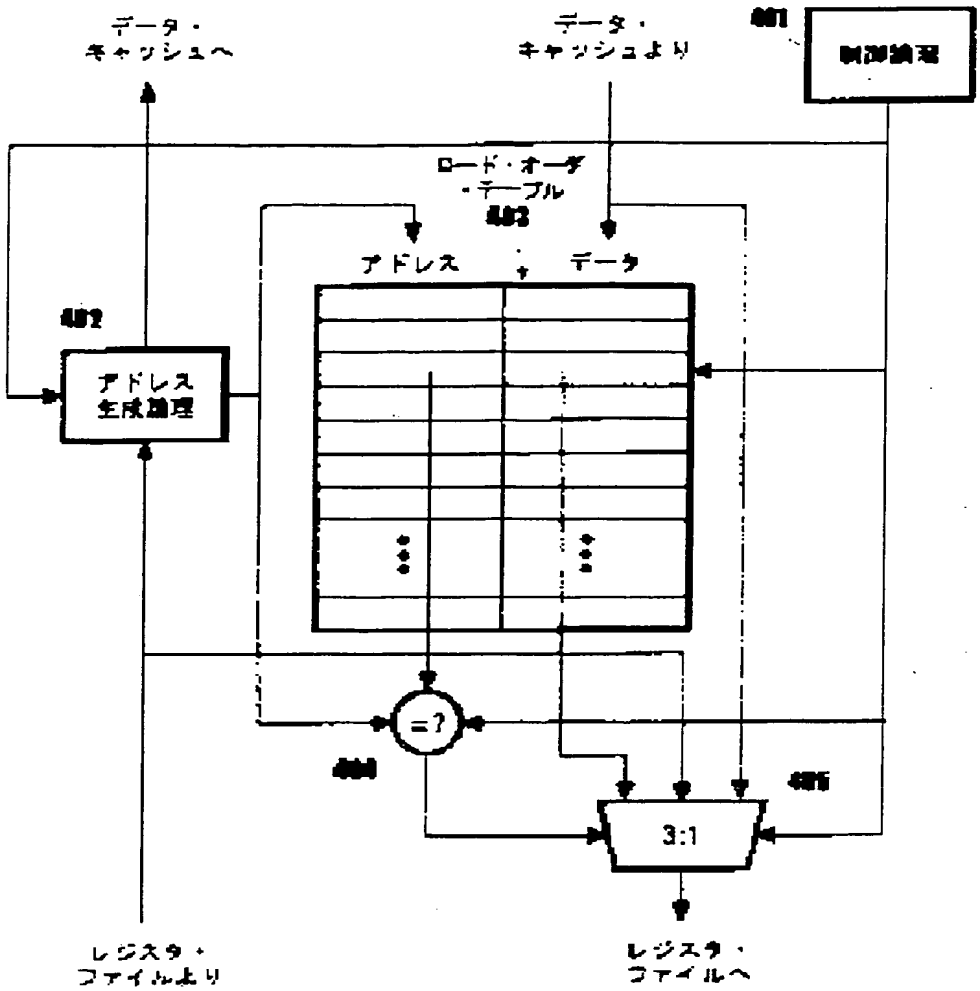
【図4】



【図7】



【図5】



【図6】